DOCUMENT RESUME

ED 329 231 IR 014 875

AUTHOR	Ellis, Edwin, Ed.
TITLE	National Educational Computing Conference Proceedings (11th, Nashville, Tennessee, June 25-27, 1990).
SPONS AGENCY	National Educational Computing Conference.
REPORT NO	ISBN-0-924667-69-9
PUB DATE	Jun 90
NOTE	358p.; For the NECC '89 Proceedings, see ED 317 190.
AVAILABLE FROM	International Society for Technology in Education, University of Oregon, 1787 Agate Street, Eugene, OR 97403-9905.
PUB TYPE	Collected Works - Conference Proceedings (021)
EDRS PRICE	MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS	Adult Learning; Class Activities; *Computer Assisted Instruction; Computer Networks; Computer Simulation; Computer Software; Disabilities; Educational Administration; Elementary Secondary Education; Higher Education; Hypermedia; Microcomputers; Multimedia Instruction; *Program Descriptions; Programing; Teacher Education; Thinking Skills; Two Year Colleges

ABSTRACT

This volume of proceedings of the 1990 National Educational Computing Conference (NECC) provides a record of the state-of-the-art in the use of computing in a variety of educational settings. Special sessions, panels, projects, 153 abstracts, and 44 papers are reported here on subjects including: elementary and secondary educational software, higher education applications, multimedia programs, hypermedia, ethics, computer education administration, interactive video, computer-assisted instruction, engineering, Logo, thinking skills, teacher education, video-based instruction, and networks. Tables and diagrams accompany some of the entries, and each of the papers contains its own list of references. An index of authors and other participants is also included. (DB)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED329231

PROCEEDINGS

National Educational Computing Conference 1990

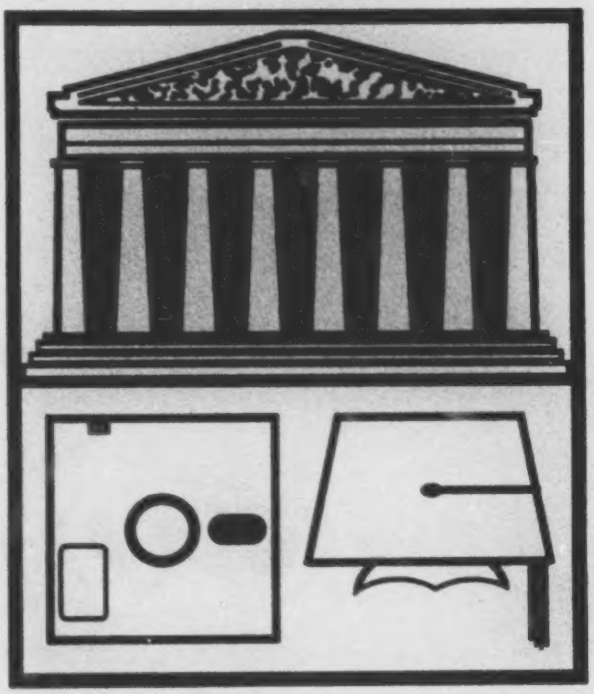
U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

☐ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality

- Points of view or opinions stated in this document do not necessarily represent official OERI position or policy

Opryland Hotel
Nashville, Tennessee
June 25-27, 1990



Hosted by Clemson University
in cooperation with
Belmont College
Mississippi State University
Murray State University

Proceedings Editor
Edwin Ellis

IR014875

"PERMISSION TO REPRODUCE THIS MATERIAL IN MICROFICHE ONLY HAS BEEN GRANTED BY

D. Moursand

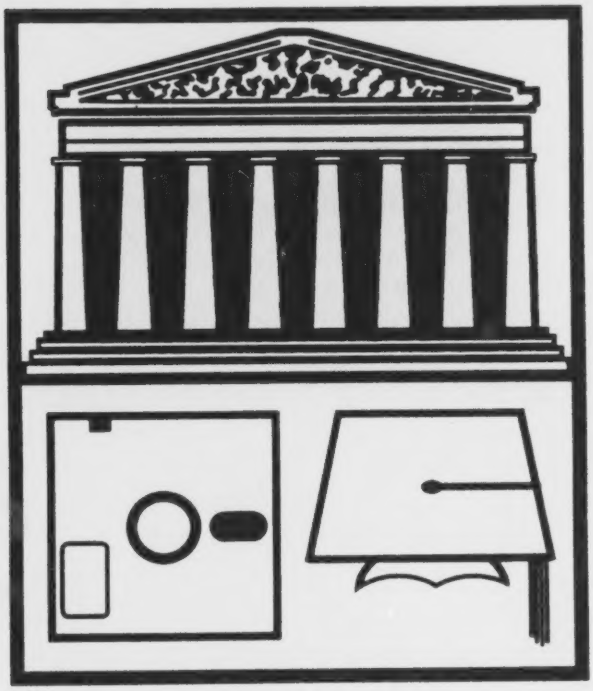
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

BEST COPY AVAILABLE

PROCEEDINGS

National Educational Computing Conference 1990

Opryland Hotel
Nashville, Tennessee
June 25-27, 1990



Hosted by Clemson University
in cooperation with
Belmont College
Mississippi State University
Murray State University

Proceedings Editor
Edwin Ellis

PROCEEDINGS OF NECC '90

International Standard Book Number: 0-924667-69-9

Copyright ©1990: NECC

Published by:
International Society for Technology in Education
University of Oregon
1787 Agate Street
Eugene, OR 97403-9905

for:
National Educational Computing Conference 1990
June 1990

FOREWORD

This volume of proceedings of the Eleventh National Educational Computing Conference (NECC '90) provides a record of the state-of-the-art in the use of computing in a variety of educational settings. NECC provides a forum for the presentation of ideas in instructional and research aspects of educational computing. Much of the information sharing in this forum is informal and spontaneous and makes the conference very worthwhile to attend in person. This proceedings contains information from the more formal portion of the conference. The papers, panels, and special sessions reported in this volume were selected on the basis of a rigorous refereeing process. The result is a program of high-quality, timely information for those of us who are concerned about educational computing.

The conference requires the time and efforts of many people. I would like to thank the members of the NECC '90 Conference Committee for their many hours of effective service to this year's conference. I would also like to thank Brenda Hart and Sharon Starks for their secretarial support and knowledgeable assistance.

The Conference Committee would like to express their appreciation to the persons who served on a variety of subcommittees.

We would all like to thank family members who patiently understood as conference matters diverted our attentions and energies. Finally we thank those who proposed sessions, presented sessions, and those who attended NECC '90.

John D. McGregor
Chair, NECC '90
Clemson University
Clemson, SC 29634

NECC'90 CONFERENCE COMMITTEE

John D. McGregor
Conference Chair
Clemson University

Helen C. Takacs
Program Chair
Mississippi State University

Patricia Finney
Local Arrangements Chair
Belmont College

Marguerite Summers
Associate Conference Chair
Murray State University

David Eldredge
Treasurer
Murray State University

Paul Katz
Exhibits and Registration
University of Oregon

Sharon Finke
Exhibits and Registration
University of Oregon

Edwin Ellis
Proceedings Editor
Mississippi State University

Richard Austing
Workshop Chair
University of Maryland

Arthur M. Riehl
Publicity Co-Chair
University of Louisville

Jim Adams
Publicity Co-Chair
ACM

NECC'90 SUBCOMMITTEES

Program Committee

Dan Brook
Mississippi State University

Judy Hankins
Middle Tennessee State Univ.

Gerry Segal
Bank Street College of Education

Gerald Engel
University of Connecticut

A. Joe Turner
Clemson University

Project Review Committee

Cornelia Brunner
Bank Street College of Education

Paul A. Reese
Ralph Bunche School

Steven T. Frantz
Scarsdale Public Schools

Paper Selection Committee

Larry Anderson
Michael Spangler
Keith Tartt
Daniel Wilson
Mississippi State University

Local Arrangements Committees

Audio/Visuals
Robert D. Sherwood
Susan M. Williams
Bill J. Corbin
Michael F. Young
Vanderbilt University

Computers
Bill Hogue
Vanderbilt University

Entertainment
W. Daniel Landes
Belmont College

Hospitality
Sharon Adams
Janell Puryear
Cherrie Farnette
University School of Nashville

Public Relations
Thom Storey
Belmont College

Signs
Joyce Blair
Frank Hughes
Delmer DeBoer
Belmont College

Volunteers
Stephen Campbell
Belmont College

Morgan Branch
Tennessee State Dept. of Education

Betty Mayberry
Rutherford County Public Schools

Woody Pigg
Metropolitan Nashville Public Schools

Workshops
David Hill
Belmont College

David Wooten
Morgan Branch
Phyllis Pardue
Jim Oakes
Philip White
Tennessee State Dept. of Education

Greg Padfield
Computer Shoppe

6

COOPERATING SOCIETIES

Association for Computing Machinery Special
Interest Groups on:
Computers and Society (SIGCAS)
Computer Science Education (SIGCSE)
Computer Uses in Education (SIGCUE)
University and College Computing Services
(SIGUCCS)

Association for Computers and the Humanities
(ACH)

Association for Small Computer Users in Education
(ASCUE)

EDUCOM

Educational Computing at Minority Institutions
(ECMI)

IEEE Computer Society
(IEEE-CS)

International Society for Technology in Education
(ISTE)

Society for Computer Simulation
(SCS)

NECC STEERING COMMITTEE, 1989-1990

Ron Anderson
ACM/SIGCAS
Mound, MN

David Brittain
Florida Department of
Education
ISTE

George Brett
University of North Carolina
ACH

Susan Friel
Co-Chairperson, NECC '89
Univerity of North Carolina

Steve Gilbert
EDUCOM
Princeton, NJ

Diana Harris
Secretary
University of Iowa

James Kerlin
Pennsylvania State University
ACM/SIGUCCS

John Lawson
Lewis Clark State College
ACM/SIGCUE

Jesse C. Lewis
Norfolk State University
ECMI

Doris K. Lidtke
Past Chairperson
Towson State University

John McGregor
Conference Chair, NECC '90
Clemson University

David Moursund
ISTE Rep. to Exec. Comm.
ISTE

Mike Mulder
University of Southwestern
Louisiana
IEEE Computer Society

Jim Poirot
Chair
University of North Texas

John Richards
Program Chair, NECC '89
BBN Labs

Nancy Roberts
Co-Chairperson, NECC '89
Lesley College
SCS

Jean Rogers
Vice Chair
Hewlett Packard

Wally Roth
Taylor University
ASCUE

Ted Sjoerdsma
Treasurer
Washington and Lee University

Helen Takacs
Program Chair, NECC '90
Mississippi State University

Joe Turner
Clemson University
ACM/SIGCSE

Association for Computers and the Humanities

The Association for Computers and the Humanities (ACH) is an international organization devoted to encouraging the development and use of computing techniques in humanities research and education. Traditionally, ACH has fostered computer-aided research in literature and language, history, philosophy, anthropology, and related social sciences as well as computer use in the creation and study of art, music, and dance. As computing applications in the humanities have developed and broadened in the 1980s, the Association has expanded it scope to include areas from word processing to computer-assisted instruction in composition, language, history, philosophy, and anthropology, as well as computational linguistics and cognitive science, which overlap increasingly with work in the area of humanities computing.

Founded in 1977, ACH is the primary professional society for humanists who are involved or interested in any aspect of humanities computing. The Association provides a forum for continuing communication about humanities computing and strives to meet the needs of those who want to gain familiarity with both existing and potential applications of computers in humanities disciplines.

The heart of ACH is its quarterly newsletter, which covers the activities of the Association and its members and includes articles on various areas within humanities computing, news of projects and conferences of interest to ACH members, and reports on activities of governmental agencies and other organizations that affect computer-aided humanities research.

Computers and the Humanities, published by Paradigm Press, is a quarterly journal devoted to scholarship in the field of humanities computing. Subscription to *CHum* is included in the ACH membership fee.

ACH sponsors the bi-annual International Conference on Computers and the Humanities (ICCH), held in odd-numbered years, which brings together scholars from around the world to report on research activities and software and hardware developments in the field. Recently, ACH began to sponsor conferences and workshops on specialized topics in humanities computing, held in even-numbered years.

ACM Special Interest Group on Computers and Society

SIGCAS is the Association for Computing Machinery's Special Interest Group on Computers and Society. With a membership of nearly 1,200, this professional group seeks to identify social issues raised by computer technology and to provide a forum for discussion on how to approach these issues.

SIGCAS publishes a quarterly newsletter, *Computers and Society*, which is a primary source of material on this topic. As a vehicle of communication for the SIGCAS membership, it includes news, comments, and articles on any societal issues raised by computing technology, One of the few periodicals on this subject, it provides a flexible and timely forum for important, evolving topics, such as data quality, employment, and intellectual property rights.

According to a recent membership survey, about 40% of SIGCAS members teach course material on computers and society. In recent years SIGCAS has organized sessions at computer conferences on topics such as computer ethics and organizational impacts of computers.

ACM Special Interest Group on Computer Science Education

SIGCSE became a special interest group of ACM in 1970. It currently consists of over 2000 members from the educational, industrial, and governmental communities interested in various aspects of computer science education. SIGCSE has goals of encouraging and assisting in the development of effective academic programs and courses in computer science and promoting research in computer science education.

The following are objectives of SIGCSE:

1. To provide a continuing forum for discussion of common problems among education and other computer scientists through organized meetings and symposia.
2. To publish a bulletin at least quarterly containing information aimed specifically at those interested in computer science education.
3. To work closely with the Education Board of ACM to insure implementation of effective education programs by the Association.

ACM Special Interest Group on Computer Uses in Education

The Special Interest Group on Computer Uses in Education (SIGCUE) of the Association for Computing Machinery (ACM) provides a forum for the discussion of ideas, methods, and policies related to all aspects of computers in the educational process. Established in 1969 its membership (over 1500 persons) comes from many countries and numerous, diverse institutions.

SIGCUE publishes a newsletter titled the SIGCUE *Outlook*. SIGCUE also sponsors and organizes technical sessions at ACM annual meetings, the National Educational Computing Conference, and other national and regional meetings of interest to its members.

Among SIGCUE's goals are (1) publishing a current, substantive bulletin, (2) cooperating with other special interest groups or educational societies to promote attention to educational computing issues, and (3) helping to bring the technical expertise within ACM to bear upon educational computing generally.

ACM Special Interest Group on University and College Computing Services

SIGUCCS provides a forum for those involved in providing computing services on a college or university campus. The topics addressed by SIGUCCS include managing campus computing, computing as it relates to the overall goals of the institution, and the state-of-the-art in various types of college and university computing services, and provides opportunities to discuss and share ideas and experiences with others.

Two annual conferences are regular activities of SIGUCCS. The Computer Center Management Symposium addresses the many aspects of managing computing on campus. This includes hardware, software, planning, finances, and personnel, to name a few. The User Services Conference deals more directly with the delivery of particular services to the higher education community. Tutorials on relevant issues are held at both conferences.

In other projects, SIGUCCS offers a Peer Review of the university computing function. Upon request of the computer center director, members of SIGUCCS will formally analyze and comment on different areas of the campus computing function. SIGUCCS also publishes a quarterly newsletter. We consider the newsletter our most important form of communication as it reaches all members and is subscribed to by numerous university computing centers. Conference proceedings are published either as separate documents or as part of the newsletter itself.

The Association of Small Computer Users in Education

The Association of Small Computer Users in Education (ASCUE) is a group of small colleges who work toward continued quality computer education and computer assistance in both administrative and academic areas. The purposes of the organization are: 1) to encourage appropriate uses of computing equipment and techniques for the improvement of its member institutions; 2) to supply its members with information on the most current computing methods; 3) to assist them in solving individual problems; 4) to cooperate with them in the utilization of the various small computers; 5) to cooperate with manufacturers, distributors, and suppliers in establishing and maintaining proper technical standards and in meeting new needs for special devices and systems.

Educational Computing in Minority Institutions

ECMI is an organization run by a steering committee representing institutions whose student body reflects a large identifiable minority population. The objectives of ECMI are:

1. Computer literary: To create among the faculty and administrators of the minority institutions an awareness and understanding of the strengths and weaknesses, uses and misuses, advantages and disadvantages, feasibility, practicability and limitations of computer applications in all aspects of society, including education.
2. Educational Computing: To narrow the gap which exists between the faculties in minority and non-minority institutions with respect to educational computing know-how and access.
3. Research Computing: To improve the computing facilities available to faculty of minority institutions for research purposes, particularly in those institutions offering graduate programs.
4. Technical assistance—consultants: To provide expert and impartial technical assistance to academic administrators of minority institutions on all phases of academic computing (instruction and research).
5. Education programs in the computer sciences: To improve the offerings of courses and degree programs in the computer sciences at minority institutions at all levels (e.g., introductory courses, minors, 2-year degree programs, 4-year degree programs, continuing education, graduate programs).
6. Computing facilities: To improve both quality and quantity of computing facilities available in minority institutions, because experience in the non-minority institutions has shown that an adequately staffed and equipped computer center for academic computing is essential to the success of previously stated objectives.
7. Direct student assistance: To increase the availability of minority staff for the computer centers and computer science education programs of minority institutions.
8. The need for a comprehensive program: To facilitate co-ordination and equitable distribution of funded activities to qualified institutions, associations, etc.

EDUCOM

EDUCOM is a nonprofit consortium of colleges, universities, and other institutions founded in 1964 to facilitate the introduction, use, and management of information technology. Through direct services and cooperative efforts, EDUCOM assists its members and provides leadership to the higher education community.

EDUCOM is funded by membership dues, service fees, and grants from foundations, corporations, and government. EDUCOM activities involve over 530 institutions and 90 corporations in the U.S. and abroad. In brief, they are: EDUCOM General Membership; the EDUCOM Consulting Group; EDUCOM Networking Activities including the Networking and Telecommunications Task Force (NTTF) and BITNIC, the BITNET Network Information Center; the EDUCOM Software Initiative; and the Corporate Associates Program.

The IEEE Computer Society

The Computer Society is the world's largest association of computing professionals, with a total membership of approximately 100,000 computer scientists, computer engineers, and allied professionals. Society membership is open to IEEE members, associate members, and student members and to non-IEEE members who qualify for affiliate membership. An affiliate member is a person who has achieved status in his or her chosen field of specialization and whose interests focus in the computing field.

Every Computer Society member receives *COMPUTER*, a peer-reviewed monthly magazine of general interest to computing professionals which also covers society news and events. Five specialized magazines and three journals are also available to society members as optional subscriptions and to nonmembers, libraries, and organizations.

Magazines published by the Computer Society include *IEEE COMPUTER GRAPHICS & APPLICATIONS, IEEE MICRO, IEEE DESIGN & TEST, IEEE SOFTWARE*, and *IEEE EXPERT*. Research-oriented journals include *IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, and *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*. The Computer Society Press publishes nonperiodical literature, including tutorial texts and conference records. The society's catalog contains approximately 800 titles which are available for purchase.

The society sponsors or cosponsors more than 100 conferences and meetings ranging from workshops and symposia with a few dozen participants to major conferences with many thousands of attendees. Over 30 technical committees offer the opportunity to interact with peers in technical specialty areas, receive newsletters, and conduct conferences and tutorials.

The Computer Society has over 100 local chapters throughout the world, and an additional 100-plus student chapters which provide the opportunity to interact with local colleagues and hear experts discuss technical issues. In addition, tutorials, educational activities, accreditation of computer science and engineering academic programs, the development of standards, and an international electronic mail network all play prominent roles in the society's activities.

International Society for Technology in Education

The International Society for Technology in Education, ISTE, is the merged society of the International Council for Computers in Education and the International Association for Computing in Education. ISTE is a non-profit educational organization, with 12,000 individual members and over 60 organization and associate members.

ISTE publishes *The Computing Teacher, Journal of Research on Computing in Education, CAELL Digest*, and books and courseware for persons interested in the instructional use of computers at the precollege level. These publications emphasize teaching about computers, teaching using computers, teacher education, and the impact of computers on curriculum.

ISTE has a substantial and growing professional outreach program. Five major components currently include:

1. Organization Affiliate Members. Members publish newsletters and/or journals, hold conferences, and directly interact with their own members.
2. Professional Staff. ISTE has a professional staff who write, edit, participate in conferences, process orders, consult by phone or mail, etc. *The Computing Teacher* is put together by a full in-house production staff.
3. Ad Hoc Committees. Such a committee created the "ICCE Policy Statement on Software Copyright" and "Code of Ethical Conduct for Computer Using Teachers."
4. Special Interest Groups. ISTE has organized special interest groups for computer coordinators, teachers of educators, computer science educators, Logo-using educators, telecommunications, and hypermedia/multimedia.
5. Independent Study Courses. ISTE offers seven independent study courses carrying graduate credit from the Oregon State System of Higher Education.

The Society for Computer Simulation

SCS is the only technical society devoted primarily to the advancement of simulation and allied technology. It has a worldwide membership and a network of regional councils that covers the United States, Canada, and the United Kingdom.

Simulation is used in every scientific and technical discipline including engineering, manufacturing, biomedical, business, and aerospace. Artificial intelligence, robotics, and CAD/ACM and simulators are areas that have been specifically recognized as important to SCS members.

There are three major SCS conferences each year. They are the SCS Western Multiconference, the SCS Eastern Multiconference, and the Summer Computer Simulation Conference (SCSC).

The National Educational Computing Conference thanks the following people for their contribution of knowledge, effort, and time as referees for the papers submitted for presentation.

Eadie Anamson
Lynne Anderson-Inman
Bill Baird
Patti Zembrosky Barkin
Bruce Barnes
Nettie R. Bartel
George M. Bass, Jr.
F. Samuel Bauer
Randy Beam
Sharon M. Bell
Anita Best
Roy K. Bhagaloo
Julie Bichteler
Rick Billstein
Gary G. Bitter
Ruthie Blankenbaker
Bill Blubaugh
Della T. Bonnette
James Bradley
Jim Cameron
Pat Campbell
Lowell Carmony
Cathy Carney
L[illegible]ase
N. John Castellan, Jr.
Donald R. Chand
Margaret Christensen
Jim Clark
Richard T. Close
Ron Cody
Betty Collis
Leslie Conery
Mary Cron
George H. Culp
Bill Davis
Herbert L. Dershem
Jean Donham Van Deusen
J. Michael Dunlap
Linda F. Ettinger
Reynolds Ferrante
Frederick T. Fink
LeRoy Finel
Dexter Fletcher
David Flowers
Michael Fry
Helen B. Frye
Judith Gersting
Allen Glenn
Harold Grossman
Cindy Hanchey
Dale Hanchey
Seymour Hanfling
Judith Harris
Janet Hartman
Brian Harvey
David Hata
Jan Hawkins
Sandee Hedetniemi
Craig Hickman
Robert Hofmeister
Mark Horney
Lawrence A. Jehn
Betty Jehn
Dale Johnson
Norman Johnson
Beverly Jones
Tony Jongejan
Ted M. Kahn
Peter Kelman
Henry Kepner
Carolyn Knox-Quinn
D. Midian Kurland
Barbara Kurshan
Annabelle Lavier
Dianna Lawyer-Brook
Laura Leventhal
Antonio M. Lopez, Jr.
William F. Lyle
Harold G. MacDermot
Gary Marchionini
Carla Mathison
Kathleen Maury
Donald H. McClain
Andrew R. Molnar
Dave Moursund
Adeline Naiman
Chris Nevison
Maggie Niess
David Olson
Janet Parker
Jim Parry
Teri Perl
Barry Pitsch
Richard Plishka
Linda G. Polin
Gerald Pollard
Hassan Pournaghshband
Gerald Rambally
Doris Ray
Paul Resta
Dick Ricketts
Robert R. Riser
Nancy Roberts
Doug Rogers
Leroy Roquemore
David H. Rose
Marian B. Rosen
Gregory C. Sales
Dean Sanders
Charlotte Scherer
Robert H. Seidman
Steve Shuller
Ellen Siegel
Robert Slotnick
Lary R. Smith
Richard Alan Smith
Dennis W. Spuck
D. Stevenson
Neal Strudler
Harriet G. Taylor
Macey Taylor
John L. Tenny
Robert A. Thomson
Leslie F. Thyberg
Samuel Tumolo
Daniel Watt
John Wedman
David Weinman
Diana M. Willis
Billy H. Wood
Sharon Yoder

Table of Contents

M1-1 HIGHER EDUCATION APPLICATIONS (PROJECTS)

1 **Teaching Calculus as a Laboratory Course**
Marcelle Bessman
1 **An Expert System for Solving Numerical Problems in Physics**
Hai Van Nguyen
2 **Coding Language vs. Authoring System (To Code or Author—That is the Question)**
Lawrence M. Kendra, Jerry Clavner

M1-2 HIGHER EDUCATION APPLICATIONS (PROJECTS)

3 **Redesigning the Computer Science Curriculum in a Liberal Arts Setting**
George T. Crocker, Susan T. Dean
3 **Faculty Expectations of Computer use in Off-Campus Courses**
John E. Cook
4 **The Role of Database Searching in Legal Education**
Anne S. Caputo, Rosalie M. Sanderson

M1-3 HIGHER EDUCATION APPLICATIONS (PROJECTS)

5 **The Interactive Classroom**
Elizabeth C. Brennan
5 **The Development and Integration of Hypermedia CAI into an Elementary Reading Program**
Randall Boone, Kyle Higgins
6 **What Do Preservice Teachers Need To Know About Technology?**
Dianne I. Novak

M1-4 DIGITAL SYSTEMS LABORATORY COURSE (SPECIAL)

7 **Digital Systems Laboratory Course**
Keith Barker

M1-5 ELEMENTARY/SECONDARY APPLICATIONS (PROJECTS)

8 **Hypermedia: Making the Learning Accountable**
Ed Coughlin
8 **Understanding the Importance of Computers in American Society**
Harriet Copel, Carol Bryne
9 **Teaching Multiple Subjects using Technology**
Frances Coleman
9 **Sound Waves: An Interdisciplinary Process Approach to Science**
Denis M. Coffey, Janice C. Kowalczyk

M1-6 MODELING: TOOLS FOR LEARNING SCIENCE AND MATH (SCS)

10 **Modeling: Tools for Learning Math and Science**
Ricardo Nemirovsky, Robert Tinker, William (Tim) Barclay, Jan Mokros

M1-7 WOMEN IN COMPUTER SCIENCE: ARE WE FAILING... (ACM/SIGCSE)

11 **Women in Computer Science: Are We Failing in Our Efforts to Recruit and Retain Them in the Field?**
Cindy Meyer Hanchey, Virginia Eaton, Sharon Bell, Susie Gallagher, Lynda Nichol

12

M1-8 COMPUTER EDUCATION RESEARCH (PROJECTS)

12 **Sports Injuries**
Kenneth E. Wright
12 **Research and Development Work of the Teacher Education Simulation Training (TEST) Group**
Jerry W. Willis, Dee Anna Willis
13 **Developmental Software Evaluation for Young Children**
Susan W. Haugland, Daniel D. Shade
13 **The MSU Computer Competency Project**
John Vinsonhaler, Hans Lee, Leighton Price, Chris Wagner

M1-9 COMPUTER EDUCATION ADMINISTRATION (PROJECTS)

14 **Computer Coordinators—Do We Have All the Answers?**
Rebecca A. Gold, Elene A. Van Noy
14 **How to Plan, Prepare, and Present a Three-day, Multi-level, Hands-on Computer Workshop**
John L. Tenny
15 **A Computer Infused Teacher Education Program**
Kathleen Maury
15 **Using Computer Simulations as part of a Field Based Model for Training Student Teacher Supervisors**
Jane H. McHaney

M4-1 ELEMENTARY & SECONDARY APPLICATIONS (PROJECTS)

16 **Fitting the Pieces Together: Successful Technology Implementation**
Hilary Cowan
16 **Project Get Ahead**
C. A. Coulter
17 **Computer Generated Biofeedback for Emotionally Disturbed High School Students**
Ned Davis
17 **The BUDDYSYSTEM Computer Project: The Curriculum-Computer Connection**
Linda M. Curry, Sharon Snellenberger

M4-2 ETHICS (PAPERS)

18 **Employing the Disabled: New Legislation, New Programs, New Technology!**
Rita Thomas Noel
23 **An Inductive Inference Approach to Plagiarism Detection in Computer Programs**
Gerard K. Rambally, Mauricio Le Sage
30 **Unethical "Computer" Behavior: Who is Responsible?**
Lawrence G. Martin

M4-4 ELEMENTARY & SCIENCE APPLICATIONS (PROJECTS)

36 **Intelligent Hypertutoring for High School Science Instruction**
John R. Bourne, Chuck Kinzer, Herman van der Molen, Jerry Haden, Dona Mularkey, Steve Schreiner, Peggy Guy, Roy Culbertson, Carlo Hyde
36 **Algorithms for Korean and Chinese Language Keyboard Input in Microcomputers**
Dennis Bilgin
37 **Computer Generated Discrepant Events—Teachable Moments in an Elementary Mathematics Classroom**
Barbara R. Biglan, Anne Stanko
37 **Project CHILD: Integrating Computers into the Elementary School Curriculum**
Sally Butzin

M4-5 MULTIMEDIA (PROJECTS)

38 **Hypercard Instructional Stacks: Jump-linear, Tree, Network, and Single-frame Examples Appropriate for the K-12 Classroom**
Michael Land
38 **Hypercard Instructional Stacks: Science and Mathematics Ideas for the K-12 Curriculum**
Steven Tipps, Michael Land
39 **The Impact of Interactive Video Presentation on High School Student Stress Management Unit**
Sharon E. Smaldino, Susan J. Koch
39 **Development and Delivery of Interactive Video and Computer Assisted Instruction at Columbia State Community College**
Stephen L. Stropes, Deanna Naddy

M4-6 ELEMENTARY & SECONDARY APPLICATIONS (PROJECTS)

40 **Building Bridges: A Report on Technology Development Sites for Speech-Language-Hearing Applications**
Paula S. Cochran, Julie H. Matherson, Frank G. Pagan, Glen L. Bull
40 **Special Touches: Adapting Computer Applications for Students with Special Needs**
Christine L. Appert
41 **Computer Assisted Instruction—It Takes a Plan**
Linda Willson, Beverly Moore
41 **Using Technology to Impact a District's Curriculum**
Brooke P. Woods

M4-7 USES FOR COMPUTERS FOR ADULT LEARNERS (PAPERS)

42 **Learning Nursing Diagnosis via Real-Time Computer-Mediated Communication**
Robert N. Higgins
48 **Interactive Multimedia Technology and the Community College: Partners in a Twenty-First Century Revolution**
Angeline Godwin Dvorak
51 **Adult Self-Directed Learning, Personal Computer Competency, and Learning Style**
Helen Barrett

M4-8 ISSUES IN TWO-YEAR COLLEGE COMPUTER EDUCATION (ACM/SIGCSE)

57 **Issues in Two-Year College Computer Education**
John Impagliazzo, Helene Chlopan, Herb Garrett, Margaret Heard

M4-9 ENGINEERING APPLICATIONS (PAPERS)

58 **Teaching a Multidisciplinary Topic Computer Integrated Manufacturing (CIM)—A Case Study**
Lee Danner, Allen E. Smith, George Stanton
62 **A Software for Computer-Aided Engineering Education**
P.J.Tatsch, F.Damiani, N.Marranghello
65 **Using Circuitmaker in a Digital Logic Class**
Terry A. Scott, Peter C. Isaacson

M4-10 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

69 **Knowledge Gateway Project**
Eric Jensen
69 **LEGO TC Logo—Fourth Grade Science Becomes Real**
JoAnn F. Karaffa
70 **Curriculum Oriented Computer Resource Guide for Middle School**
Mary Grace Jaeger, Cheryl L. Hepp
70 **Hypercard: Reinventing the School**
Joseph Hofmeister, Joyce Rudowski

M4-11 EARLY CHILDHOOD (PAPERS)

71 Successes Despite Extremely Limited Computing Time in a Primary Classroom
Sandee Hedetniemi, Helen Robinson
77 The Willingness of Rural Iowa Educators to Participate In Computer Teleconferencing
Lowell Monke, Sharon E. Smaldino
81 Collaboration and Competition: An Innovative Approach to Computer Learning
Ms. Sydney Bennett, Ms. Judy Kane, Dr. Pamela Munz

T1-2 MANAGING & FUNDING COMPUTING IN MINORITY INSTITUTIONS (ECMI)

85 Managing and Funding Computing in Minority Institutions
Jesse Lewis, William Lupton, Larry Oliver

T1-3 PEOPLE, LOVE, AND COMPUTER IN EDUCATION... (SPECIAL)

86 People, Love, and Computer in Education: A Perspective on Ten Years' Involvement
Betty Collis

T1-4 HIGHER EDUCATION (PAPERS)

87 Merging Object-Oriented Development Into Introductory Computer Science Courses
Martha McCormick, Ronald White
91 Historical Perturbation and Reconstruction: A Computers and Society Teaching Method
William J. Joel
94 Thinking Skills, Databases, and Nutrition
Gertrude W. Abramson

T1-5 THE COMPUTER AS A TOOL (PAPERS)

97 Computer Use for Newswriting Evaluation and Feedback
William Edward Smith
103 Integration of Product Development Cycle and Component Display Theory to Create Computer Aided Instruction
Robert Perkins
109 Teaching and Learning with Computer Based Instruction: Findings from New York City's Computer Pilot Program
Karen Swan, Marco Mitrani, Frank Guererro, John Schoener

T1-6 MULTIMEDIA, HYPERMEDIA, AND THE WRITING PROCESS (SPECIAL)

118 Multimedia, Hypermedia, and the Writing Process
Stephen Marcus

T1-7 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

119 Exploring Art and Technology Using LEGO TC Logo
Cathy Helgoe, Eadie Adamson
119 Integrating the Computer, Writing Development, and Self-Esteem in the Middle School Child
Mary S. Haverfield
120 New Initiatives in Technology Education
Martha Hancock
120 A Hard Drive at Risk: An Apple "Equal Time Grant" Winner
Carol Goltz

T1-8 WHAT EXACTLY DO WE MEAN BY "LOGO-LIKE" (ISTE)

121 What Exactly Do We Mean By "Logo-Like"?
Gary S. Stager, Glen Bull, Brian Silverman, Sharon Yoder, Marian B. Rosen, Eadie Adamson

T1-9 INQUIRY, COMPUTER, AND THE MATHEMATICS CLASSROOM (SCS)

122 **Inquiry, Computer, and the Mathematics Classroom**
Nancy Roberts, Richard Carter, John Richards, Cornelia Tierney

T1-10 ACCREDITATION ISSUES DEALING WITH COMPUTER ETHICS (SPECIAL)

123 **Accreditation Issues Dealing with Computer Ethics**
Terrell Ward Bynum, Joseph Turner, Walter Maner

T1-11 BACK TO THE FUTURE... (SCS)

124 **Back to the Future: Telecommunications Beyond Motivation**
Robert Spielvogel, Al Rogers, Robert Gehorsam, Martin Harris, Ellen Chaffin, Bobby Kurshan, Martin Engel, Linda Roberts

T1-12 NATIONAL GEOGRAPHIC'S KIDS NETWORK (SPECIAL) Linda Roberts Engel

125 **National Geographic's Kids Network**
Marti Cantrell

T3-1 VALUES IN THE COMPUTER SCIENCE CURRICULUM (SPECIAL)

126 **Values in the Computer Science Curriculum**
Terrell Ward Bynum, Keith Miller

T3-2 THE ROLE OF TECHNOLOGY IN THE RESTRUCTURING... (ISTE)

127 **The Role of Technology in the Restructuring of Schools**
Tony Jongejan, Alan November, Bob Pearlman, Doris Ray

T3-3 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

128 **Macintosh Lab for the Visually and Hearing Impaired**
Janet Fleharty
128 **High School by Satellite in Kentucky: Live and Interactive!**
Leslie Flanders
129 **Learning "The Write Stuff" Through Computer Technology**
Debra Freedman, Jerry Stimmel

T3-4 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

130 **Using LogoWriter to Enhance Learning of Elementary French**
Eadie Adamson
130 **If This is Tuesday, This must be Belgium...Enhancing Foreign Language/Computer Skills through the Use of Spanish and French LogoWriter/Telecommunications**
Leon Foster
131 **Using Technology to Enrich the Whole Language Classroom**
Peggy Healy Stearns

T3-5 ESTABLISHING ACCREDITATION GUIDELINES... (ISTE)

132 **Establishing Accreditation Guidelines in Technology for Teacher Education Programs**
Lajeane G. Thomas, C. Dianne Martin, Gary Bitter, Sally Sloan, Don Knezek

T3-6 REFLECTIONS ON A DECADE OF LOGO USE IN SCHOOLS (ISTE)

133 **Reflections on a Decade of Logo Use in Schools**
Gary S. Stager, Marian B. Rosen, Michael I. Tempel, Linda Polin, E. Paul Goldenberg, Tom Lough

T3-8 ADMINISTRATIVE APPLICATIONS (PAPERS)

134 **An Analysis of Administrative Computer Use By Secondary Principals In Kentucky**
D. W. Witten, M. D. Richardson, R. L. Prickett
140 **Principals' Perspectives on Computerized Student Information Management**
Dr. Peter Wright

T3-9 ELEMENTARY SOFTWARE FOR THE NEXT GENERATION (ISTE)

147 **Elementary Software for the Next Generation**
Barbara Kurshan, Beverly Hunter, Peter Kelman, Marge Kosel Cappo, John Richards

T3-10 VISION TEST: WHAT OUR SCHOOLS COULD BE LIKE (ISTE)

148 **Vision Test: What Our Schools Could Be Like**
Robert D'Ambrosio, Ludwig Braun

T4-1 NSF PROGRAMS IN ETHICS AND VALUE STUDIES (SPECIAL)

149 **NSF Programs in Ethics and Value Studies**
Terrel Ward Bynum, Rachelle Hollander

T4-2 QUALITY OF LIFE AND COMPUTER EDUCATION (ACM/SIGCAS)

150 **Quality of Life and Computer Education**
Ronald E. Anderson, C. Dianne Martin, Barbara Kurshan, Richard Rosenberg

T4-3 TEACHER EDUCATION (PAPERS)

151 **Integrating Technology into the Classroom Teachers...The Important Piece in the Puzzle**
Kam Matray
155 **Computing Across the Curriculum: An Integral Part of Preservice Teacher Education**
Nancy Cunniff
160 **Teacher Support for Technology Integration: What Worked; What Didn't**
Linda G. Polin

T4-4 ELEMENTARY & SECONDARY APPLICATIONS (PROJECTS)

167 **Make It Happen—Then Make It Stick**
Kathryn Kowalczyk
167 **Applesources: Extending and Integrating the Classroom into the 21st Century**
Joyce Morris
168 **Algorithmic Thinking: A Key to Problem Solving**
Anita Marie Stacy

T4-5 INSTRUCTIONAL COMPUTING (PAPERS)

169 **Alternatives to Integrated Instructional Systems**
Peter Kelman
177 **Working Together: Increasing the Effectiveness of Collaborative Computer Based Learning Groups**
Judi Repman
182 **Embedding Learning About Women's Health Care in Computer-based Simulations**
Joyce E. White

T4-6 HIGHER EDUCATION APPLICATIONS (PROJECTS)

187 **Conceptual Understanding of Software Packages**
Danielle R. Bernstein
187 **Using the Modular Approach to User Manuals in Teaching Documentation**
W. Steve Anderson
188 **Interactive High Speed Computer Classroom**
Richard A. Alo', Carol Jones Vobach
188 **Computers and the Visual Artist**
Marc J. Barr

T4-7 PRINCIPLES FOR ESTABLISHING A STATE... (ISTE)

189 **Principles for Establishing a State Public Education Network**
Connie Stout, Alex Belous, Judi Harris, Paul Resta

T4-9 COMPUTERS AS LEARNING TOOLS ACROSS THE CURRICULUM (SPECIAL)

190 **Computers as Learning Tools Across the Curriculum: Vanderbilt University's Experience with an Electronic Classroom**
William F. Hogue

T4-10 REASONING SKILLS (PAPERS)

191 **Teaching Analogical Reasoning through Guided Logo Programming**
Neal Grandgenett, Ann Thompson
202 **Functions, Transformations, and Mapping via Computer Graphics**
Mary Dwyer Wolfe, Tom Brieske, Hiram Johnston
210 **Characteristics of a Logo Programming Culture**
James Dunne

T4-11 MULTIMEDIA (PROJECTS)

217 **Opening Windows to Technology**
Faye Wilmore
217 **Multimedia in Teacher Education**
Ronald J. Abate
218 **The Florida-England Connection: A Telecommunications Project for Kids**
M. D. Roblyer

W1-1 NEW TRENDS IN PROGRAMMING (PAPERS)

219 **The Use of Hypertext Methodologies in Teaching Pascal**
Linda H. Rosenberg, Charles Nicholas
225 **Teaching Programming Using the Karel the Robot Paradigm Realized with a Conventional Language**
Roland H. Untch
230 **Turtle Graphics Can Enhance an Introductory Programming Course**
Dean Sanders

W1-2 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

234 **Kid Link: An Elementary Curriculum Telecommunications Project**
Kendall Stall, Virginia Lawson, Dolores Alexander
234 **LearningSphere 2008: A "Slice" of Tomorrow's School**
Eileen H. Steele, Nancy A. S. Miller
235 **The Regional Database: Critical Thinking in Social Science**
Robert H. Summerville, Christine E. Drew
235 **The Satellite Office: An Integrated Telecommunications Project**
Connie Stout

W1-3 MULTIMEDIA/NEWORKING (PAPERS)

236 **Connectivity—From the Macintosh to Other Computer Networks**
Tara Gilmore Shlosman, Dr. Virginia Eaton
241 **Incorporating Advanced Technologies into the Undergraduate Introductory Computing Course for Non-Majors**
Daniel Farkas
245 **Chaplin:** Combining HyperCard and Prolog Inference for Hypermedia-based Instruction
Víctor M. Mendoza-Grado

W1-5 VIDEO-BASED INSTRUCTION AT PEABODY COLLEGE... (SPECIAL)

249 **Using Videodisc Macrocontexts to Enhance Middle School Problem Solving Instruction**
Michael Young, Nancy J. Vye, Susan Williams, James Van Haneghan, Linda Barron, John D. Bransford, Susan R. Goldman
249 **Effects of Multimedia to Enhance Writing Ability**
Charles Kinzer, Ted Hasselbring, Connie Schmidt, Laurie Meltzer
250 **Effects of Videodisc Macrocontexts on Comprehension and Composition of Causally-Cohesive Stories**
Victoria Risko, Charles Kinzer, Nancy Vye, Deborah Rowe
250 **Using Videodisc-Based Instruction to Develop Computational and Conceptual Understanding in Elementary School Mathematics**
Robert D. Sherwood, Ted S. Hasselbring, E. Jean March, Jill C. Mertz

W1-5 ELEMENTARY & SECONDARY APPLICATIONS (PROJECTS)

251 **A Writing Process for the 21st Century: Ideas for Using Desktop Publishing in the Elementary and Middle School**
Sherah B. Nelson, Brevard Williams, III
251 **Music and Computers for Children with Special Needs**
Nancy A. Norman
252 **An American Experience**
Albert Ortiz

W1-6 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

253 **High Tech at Tech High**
Joyce Perkins, Jim Hartman
253 **Technology Integration in Rural Settings: A Bridge Between Computer Applications and Local Curriculum**
Jim Parry
254 **Belridge School District: Planning for Technology Integration DACOTT 21/20 (District and Community of Tomorrow Today)**
A Community of Learners' Model Technology Project
Gary Peterson, Cyndy Everest-Bouch
255 **Self-Made Database Projects for the Secondary School**
Donald Pratt

W1-7 HIGHER EDUCATION APPLICATIONS (PROJECTS)

256 **Equal Time for Teachers: A Helping of Apple's Pie**
Cathy Thurston
256 **Successful Staff Development**
Michael I. Tempel
257 **Classroom Applications of Expert Systems**
Roy Tamashiro

W1-8 HIGHER EDUCATION APPLICATIONS (PROJECTS)

258 Middle School Math in a Technological Society
Janet Parker
258 The Right to Succeed—A Progress Report on the Stevens Institute of Technology's Science and Technology Enrichment Project (STEP) for Disadvantaged Students Featuring LEGO TC Logo
Gary S. Stager
259 An Experiment with Peer Tutoring in a Microcomputer Applications Course
Janet Smith
259 Curriculum and Philosophy of the Master Degree in Computers in Education at Inter American University of Puerto Rico
Eduardo Rivera

W1-9 A PROJECT-ORIENTED COURSE IN SOFTWARE ENGINEERING (SPECIAL)

260 A Project-Oriented Course in Software Engineering
Gary Ford

W1-10 FAIRS, CONFERENCES, CONTESTS... (SPECIAL)

261 Fairs, Conferences, Contests, and Activities for Kids
Doris M. Carey

W2-1 PROGRAMMING (PAPERS)

262 Using Modula-2's Module Concept as the Basis of a Team Project in the First Computer Science Course
Peter Isaacson, Terry Scott
267 Learning to Program with Structure Editing: An Update and Some Replications
Dennis R. Goldenson
273 Logo before Prolog
Alan C. Cutting

W2-2 NSF FUNDING OPPORTUNITIES

276 NSF Funding Opportunities
Anita J. La Salle

W2-3 TRUE BASIC AS A LANGUAGE FOR COMPUTER SCIENCE (SPECIAL)

277 True BASIC as a Language for Computer Science
Thomas E. Kurtz

W2-4 MULTIMEDIA PROJECTS ELEMENTARY (PROJECTS)

278 Using CD-ROM for Research in the Media Center and Classroom
Michael K. Russ, Janet Leistner
278 Integrating Computer Skills into a Transitional Education Program
Anthony H. Robinson
279 Team Logo Meets Newton and Maxwell
Camille Minichino, Stephen C. Sesko

W2-5 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

280 **A Transformed Learning Center: Using Technology to Restructure Schools**
Sara O. Schoenefeld
280 **The Computer-Greenhouse Effect**
Alan Solomon
281 **Using Dazzle Draw with Behavior Disordered Students**
Sandra Smith
281 **Telecommunications for Critical Inquiry: The Homeless Around the Continent**
Lynne Schrum

W2-6 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

282 **Indiana's Buddy System Project**
Brenda Raker
282 **European Schools Project**
Henk Sligte
283 **Maryland Education Project: Computer Integrated Instruction Training**
Barbara Reeves, Patricia Mullinix
283 **Chapter 1 and Technology: De-Papering Chapter 1**
Jean C. Reynolds, David H. Berenson

W2-7 WRITING WITH COMPUTERS (PAPERS)

284 **Dynamic Writing: A Glimpse into the Future**
Ihor Charischak
287 **Computer Labs in College Residences: Opportunities for Informal Learning—Especially for Women and Minorities**
Pamela Freyd, Christopher Dennis, Thomas Kinsella
295 **Improving the Writing Ability Of Middle Level Students Through a Role-Taking Experience Using Computers**
Jane D. Steelman

W2-8 MULTIMEDIA (PROJECTS)

301 **Learning Japanese Literature Through a "Hyperfilm"**
Hiroo Saga, Yasuki Hamano, Masaaki Hagino
301 **Sights, Sounds, Science, Stacks: Hyperstudio and Interactive Video**
E. Byron Rogers, Lynn M. Mason
302 **The Effects of Progressive Levels of Interactivity in an Interactive Video Program on Learning**
Margaret Lynn Bailey, Jackson Byars
302 **Development and Use of Interactive Video CAI to Prepare Community College Students for Minimal-Competency Examinations in Mathematics**
Zan Tamar Bailey, Cynthia A. Elliott

W2-9 COMPUTER EDUCATION RESEARCH (PROJECTS)

303 **Activities to Motivate Women to Use Technology in the Humanities**
Vanessa Evans Huse
303 **Computer Learning in Adults: Using Kolb's Adult Learning Styles Inventory to Assess Adult Computer Learning**
Jill H. Ellsworth
304 **Ethical Insight of Undergraduates: The Influence of Family, Friends, Religion, and Previous Education on Freshman Understanding of Ethics**
Janet M. Cook

W2-10 EXTENDING PRESERVICE TEACHER PREPARTION... (ISTE)

305 **Extending Preservice Teacher Preparation Programs with Distance Education Technologies**
Judi Harris, Rhys Gwyn, David Bell

W2-11 HIGHER EDUCATION (PAPERS)

306 **A Study of the Use of Interactive Videodisc Technology to Present Aural Tests to College Music Appreciation Students**
Ernest Woodruff, Phillip Heeler
310 **Using "Teaching Information Systems" to Reduce the Barriers between Teaching and Research.**
Russell L. Shackelford, Albert N. Badre

W2-12 COMPUTER SCIENCE AND ENGINEERING CURRICULUM... (ACM/SIGCSE & IEEE-CS)

316 **Computer Science and Engineering Curriculum Recommendations: A Report from the Joint ACM/IEEE-CS Task Force**
Allen B. Tucker, Keith Barker, Doris K. Lidtke, A. Joe Turner

W3-1 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

317 **Magic Characters Game**
John T. Taylor, Marcelle Bessman
317 **The Other Side: Desktop Publishing for At-Risk Students**
Kathleen A. Sutphen
318 **Computers, Curricula, and Special Learners**
Gary G. Bitter, Mary Hatfield, Janie Wilson, Ruthie Blankenbaker, Shelley B. Wepner
318 **Linking with Europe**
Joanne Troutner

W3-2 INCENTIVE FUNDING IN TECHNOLOGY (SPECIAL)

319 **Incentive Funding in Technology: The New Jersey Model for Stimulating Improvement in Undergraduate Education**
James Kinnamon

W3-3 THE COMPUTER, A DEVELOPING CHILD'S TOOL OF VISION (SPECIAL)

320 **The Computer, A Developing Child's Tool of Vision**
Nancy Scali

W3-4 TOMORROW'S MUTLTIMEDIA TECHNOLOGY IN... (SPECIAL)

321 **Tomorrow's Multimedia Technology in Today's Classroom Media Integration Centre**
Brent Wilson, Dale Henderson

W3-5 HIGHER EDUCATION APPLICATIONS (PROJECTS)

322 **Implementation of a Biometry and Medical Computing Course**
Leo M. Harvill, John H. Kalbfleisch
322 **On Beyond Word Processing: Computer Supported Writing in the Elementary School Classroom**
D. Midian Kurland , Charles W. Fisher
323 **Computer Animation Software Libraries**
William J. Joel
323 **Narrative Engines: Modeling the Application of Literary Theory**
Peter Havholm, Larry L. Stewart

W3-6 HIGHER EDUCATION APPLICATIONS (PROJECTS)

324 **Self-Made Database Projects for University Level Education**
Mary G. Harris, Donald L. Pratt
324 **Managing Campus-Wide Information Systems**
Timothy J. Foley
325 **Changing the Precalculus Curriculum using Mathematica**
Beva Eastman
325 **The Evaluative Imaging of Mental Models: Visualizing Cognitive Reality**
Chris Dede

W3-7 RESEARCH (PROJECTS)

326 **Rural Teachers a Year Later: Providing Continuing Support in Educational Computing**
Diane McGrath
326 **The Use of Computer Assisted Instruction in High School Courses Offered via Satellite**
Susan M. McClelland, Linda Bennett, William D. Cole
327 **Educational Technology Instruction and Integration Program for School of Education Methodology Faculty**
Nancy J. Martin, Susan M. Cooper-Shoup

W3-8 A TUTORIAL INTRODUCTION TO NEURAL NETWORKS (SCS)

328 **A Tutorial Introduction to Neural Networks**
A. Martin Wildberger

W3-9 INTERACTIVE ESCHER... (SPECIAL)

329 **Interactive Escher—Introductory Transformational Geometry Concepts in a Museum Environment**
Gerry Segal

W3-10 EDUCATIONAL ELECTRONIC NETWORKS... (SPECIAL)

330 **Educational Electronic Networks: In Theory and in Practice**
Hugh Mehan, Bertrum Bruce, Al Rogers, James Levin, Michael Waugh, Haesun Kim, Cathy Thurston, Naomi Miyake, Barbara Brehm, Gin-Fon Ju, Margaret Riel, Denis Newman, Shelley Goldman, Seth Chaiklin, Cecilia Lenk

W3-11 COMPUTING, INTELLECTUAL PROPERTY AND EDUCATION... (EDUCOM)

331 **Computing, Intellectual Property and Education—What are the Limits? Where are the Problems?**
Frank W. Connolly, Steve Gilbert, Peter Lyman

Index of Authors and Other Participants

Abate, Ronald J. 217
Abramson, Gertrude W. 94
Adamson, Eadie 119,121,130
Alexander, Dolores 234
Alo', Richard A. 188
Anderson, Ronald E. 150
Anderson, W. Steve 187
Appert, Christine L. 40
Badre, Albert N. 310
Bailey, Margaret Lynn 302
Bailey, Zan Tamar 302
Barclay, William (Tim) 10
Barker, Keith 7,316
Barr, Marc J. 188
Barrett, Helen 51
Barron, Linda 249
Bell, David 305
Bell, Sharon 11
Belous, Alex 189
Bennett, Linda 326
Bennett, Ms. Sydney 81
Berenson, David H. 283
Bernstein, Danielle R. 187
Bessman, Marcelle1, 317
Biglan, Barbara R. 37
Bilgin, Dennis 36
Bitter, Gary G. 132,318
Blankenbaker, Ruthie 318
Boone, Randall 5
Bourne, John R. 36
Bransford, John D. 249
Braun, Ludwig 148
Brehm, Barbara 330
Brennan, Elizabeth C. 5
Brieske, Tom 202
Bruce, Bertrum 330
Bryne, Carol 8
Bull, Glen L. 40,121
Butzin, Sally 37
Byars, Jackson 302
Bynum, Terrell Ward 123,126,149
Cantrell, Marti 125
Cappo, Marge Kosel 147
Caputo, Anne S 4
Carey, Doris M. 261
Carter, Richard 122
Chaffin, Ellen 124
Chaiklin, Seth 330
Charischak, Ihor 284
Chlopan, Helene 57
Clavner, Jerry 2
Cochran, Paula S. 40
Coffey, Denis M. 9
Cole, William D. 326
Coleman, Frances 9
Collis, Betty 86
Connolly, Frank W. 331
Cook, Janet M. 304
Cook, John E. 3
Cooper-Shoup, Susan M. 327
Copel, Harriet 8
Coughlin, Ed 8
Coulter, C. A. 16
Cowan, Hilary 16
Crocker, George T. 3
Culbertson, Roy 36
Cunniff, Nancy 155
Curry, Linda M. 17
Cutting, Alan C. 273
D'Ambrosio, Robert 148
Damiani, F. 62
Danner, Lee 58
Davis, Ned 17
Dean, Susan T. 3
Dede, Chris 325
Dennis, Christopher 287
Drew, Christine E. 235
Dunne, James 210
Dvorak, Angeline Godwin 48
Eastman, Beva 325
Eaton, Dr. Virginia 11,236
Elliott, Cynthia A. 302
Ellsworth, Jill H. 303
Engel, Martin 124
Everest-Bouch, Cyndy 254
Farkas, Daniel 241
Fisher, Charles W. 322
Flanders, Leslie 128
Fleharty, Janet 128
Foley, Timothy J. 324
Ford, Gary 260
Foster, Leon 130
Freedman, Debra 129
Freyd, Pamela 287
Gallagher, Susie 11
Garrett, Herb 57
Gehorsam, Robert 124
Gilbert, Steve 331
Gold, Rebecca A. 14
Goldenberg, E. Paul 133
Goldenson, Dennis R. 267
Goldman, Shelley 330
Goldman, Susan R. 249
Goltz, Carol 120
Grandgenett, Neal 191
Guererro, Frank 109
Guy, Peggy 36
Gwyn, Rhys 305
Haden, Jerry 36
Hagino, Masaaki 301
Hamano, Yasuki 301
Hanchey, Cindy Meyer 11
Hancock, Martha 120
Haneghan, James Van 249

Harris, Judi 189,305
Harris, Martin 124
Harris, Mary G. 324
Hartman, Jim 253
Harvill, Leo M. 322
Hasselbring, Ted S. 249,250
Hatfield, Mary 318
Haugland, Susan W. 13
Haverfield, Mary S. 119
Havholm, Peter 323
Heard, Margaret 57
Hedetniemi, Sandee 71
Heeler, Phillip 306
Helgoe, Cathy 119
Henderson, Dale 321
Hepp, Cheryl L. 70
Higgins, Kyle 5
Higgins, Robert N. 42
Hofmeister, Joseph 70
Hogue, William F. 190
Hollander, Rachelle 149
Hunter, Beverly 147
Huse, Vanessa Evans 303
Hyde, Carlo 36
Impagliazzo, John 57
Isaacson, Peter C. 65,262
Jaeger, Mary Grace 70
Jensen, Eric 69
Joel, William J. 91,323
Johnston, Hiram 202
Jongejan, Tony 127
Ju, Gin-Fon 330
Kalbfleisch, John H. 322
Kane, Ms. Judy 81
Karaffa, JoAnn F. 69
Kelman, Peter 147,169
Kendra, Lawrence M. 2
Kim, Haesun 330
Kinnamon, James 319
Kinsella, Thomas 287
Kinzer, Charles 36,249,250
Knezek, Don 132
Koch, Susan J. 39
Kowalczyk, Janice C. 9
Kowalczyk, Kathryn 167
Kurland , D. Midian 322
Kurshan, Barbara 147,150
Kurshan, Bobby 124
Kurtz, Thomas E. 277
Land, Michael 38,38
Lawson, Virginia 234
Lee, Hans 13
Leistner, Janet 278
Lenk, Cecilia 330
Levin, James 330
Lewis, Jesse 85
Lidtke, Doris K. 316
Lough, Tom 133
Lupton, William 85
Lyman, Peter 331
Maner, Walter 123
March, E. Jean 250
Marcus, Stephen 118
Marranghello, N. 62
Martin, C. Dianne 132,150
Martin, Lawrence G. 30
Martin, Nancy J. 327
Mason, Lynn M. 301
Matherson, Julie H. 40
Matray, Kam 151
Maury, Kathleen 15
McClelland, Susan M. 326
McCormick, Martha 87
McGrath, Diane 326
McHaney, Jane H. 15
Mehan, Hugh 330
Meltzer, Laurie 249
Mendoza-Grado, Víctor M. 245
Mertz, Jill C. 250
Miller, Keith 126
Miller, Nancy A. S. 234
Minichino, Camille 279
Mitrani, Marco 109
Miyake, Naomi 330
Mokros, Jan 10
Molen, Herman van der 36
Monke, Lowell 77
Moore, Beverly 41
Morris, Joyce 167
Mularkey, Dona 36
Mullinix, Patricia 283
Munz, Dr. Pamela 81
Naddy, Deanna 39
Nelson, Sherah B. 251
Nemirovsky, Ricardo 10
Newman, Denis 330
Nguyen, Hai Van 1
Nichol, Lynda 11
Nicholas, Charles 219
Noel, Rita Thomas 18
Norman, Nancy A. 251
Novak, Dianne I. 6
November, Alan 127
Noy, Elene A. Van 14
Oliver, Larry 85
Ortiz, Albert 252
Pagan, Frank G. 40
Parker, Janet 258
Parry, Jim 253
Pearlman, Bob 127
Perkins, Joyce 253
Perkins, Robert 103
Peterson, Gary 254
Polin, Linda G. 133,160
Pratt, Donald L. 255,324
Price, Leighton 13
Prickett, R. L. 134
Raker, Brenda 282

Rambally, Gerard K. 23
Ray, Doris 127
Reeves, Barbara 283
Repman, Judi 177
Resta, Paul 189
Reynolds, Jean C. 283
Richards, John 122,147
Richardson, M. D. 134
Riel, Margaret 330
Risko, Victoria 250
Rivera, Eduardo 259
Roberts, Linda 124
Roberts, Nancy 122
Robinson, Anthony H. 278
Robinson, Helen 71
Roblyer, M. D. 218
Rogers, Al 124,330
Rogers, E. Byron 301
Rosen, Marian B. 121
Rosen, Marian B. 133
Rosenberg, Linda H. 219
Rosenberg, Richard 150
Rowe, Deborah 250
Rudowski, Joyce 70
Russ, Michael K. 278
Saga, Hiroo 301
Sage, Mauricio Le 23
Salle, Anita J. La 276
Sanders, Dean 230
Sanderson, Rosalie M. 4
Scali, Nancy 320
Schmidt, Connie 249
Schoenefeld, Sara O. 280
Schoener, John 109
Schreiner, Steve 36
Schrum, Lynne 281
Scott, Terry A. 65,262
Segal, Gerry 329
Sesko, Stephen C. 279
Shackelford, Russell L. 310
Shade, Daniel D. 13
Sherwood, Robert D. 250
Shlosman, Tara Gilmore 236
Silverman, Brian 121
Sligte, Henk 282
Sloan, Sally 132
Smaldino, Sharon E. 39,77
Smith, Allen E. 58
Smith, Janet 259
Smith, Sandra 281
Smith, William Edward 97
Snellenberger, Sharon 17
Solomon, Alan 280
Spielvogel, Robert 124
Stacy, Anita Marie 168
Stager, Gary S. 121,133,258
Stall, Kendall 234
Stanko, Anne 37
Stanton, George 58
Stearns, Peggy Healy 131
Steele, Eileen H. 234
Steelman, Jane D. 295
Stewart, Larry L. 323
Stimmel, Jerry 129
Stout, Connie 189,235
Stropes, Stephen L. 39
Summerville, Robert H. 235
Sutphen, Kathleen A. 317
Swan, Karen 109
Tamashiro, Roy 257
Tatsch, P.J. 62
Taylor, John T. 317
Tempel, Michael I. 133,256
Tenny, John L. 14
Thomas, Lajeane G. 132
Thompson, Ann 191
Thurston, Cathy 256,330
Tierney, Cornelia 122
Tinker, Robert 10
Tipps, Steven 38
Troutner, Joanne 318
Tucker, Allen B. 316
Turner, A. Joe 316
Turner, Joseph 123
Untch, Roland H. 225
Vinsonhaler, John 13
Vobach, Carol Jones 188
Vye, Nancy J. 249,250
Wagner, Chris 13
Waugh, Michael 330
Wepner, Shelley B. 318
White, Joyce E. 182
White, Ronald 87
Wildberger, A. Martin 328
Williams III, Brevard 251
Williams, Susan 249
Willis, Dee Anna 12
Willis, Jerry W. 12
Willson, Linda 41
Wilmore, Faye 217
Wilson, Brent 321
Wilson, Janie 318
Witten, D. W. 134
Wolfe, Mary Dwyer 202
Woodruff, Ernest 306
Woods, Brooke P. 41
Wright, Dr. Peter 140
Wright, Kenneth E. 12
Yoder, Sharon 121
Young, Michael 249

Teaching Calculus as a Laboratory Course

Marcelle Bessman
Frostburg State University

Abstract

In the Fall 1989 semester I taught two Calculus I sections and one Calculus III section as laboratory science courses, using MicroCalc by Harley Flanders for both demonstrations and as an "exploration and experimentation tool." This semester I am using the software to teach Calculus I, Calculus for Applications II, and to a lesser extent, College Algebra.

I use the computer in the classroom to demonstrate various concepts, such as function, limit, tangent, derivative, continuity, asymptotic behavior, L'Hospital's Rule, and infinite sequences or series. Laboratory assignments are used to enhance these demonstrations. The assignments are designed to provide guidance for exploration and promote student "what if" responses. Students are required to submit written laboratory reports with description, observations, and summary of experiment in essay form. Students are encouraged to work in groups of two or three persons.

I have been experimenting with the development of interactive computer assisted instruction using the development software, *Linkway*. "Hot spots" in the teaching module allow students to select to view an animated demonstration of a concept, obtain more information about the concept, or experiment with the concept using MicroCalc or other mathematical software.

This presentation will contain examples of classroom demonstrations, laboratory assignments, and student laboratory reports. The results of this project, including student evaluation of the experience as well as commentary of other members of the faculty, will be discussed. In addition, the advantages and disadvantages of using MicroCalc and some alternative packages that were considered will be discussed.

An Expert System for Solving Numerical Problems in Physics

Hai Van Nguyen
Jefferson Community College
Louisville, Kentucky

Abstract

This tutorial program written in Turbo Prolog is intended for slow students in college physics. It uses the logic-based inference method to help them solve numerical problems.

When the program is run, a knowledge base is loaded. An opening menu prompts the user to choose either consultation or viewing a worked problem from a database. If consultation is chosen, a dialogue takes place between the user and the expert system. At the outset, the user should have a clear idea of the information given in the problem and what is being asked. The first step is to enter a specific goal the system is to achieve in the form of a quantity to be found. The inference engine then looks through the knowledge base clauses to identify the existence or absence of the pertinent data values. As the consultation unfolds, the system will ask the user about the information given in the problem. Based on the user's responses, the expert system suggests a solution method or sends the message "Sorry I can't help you."

Coding Language vs. Authoring System
(To Code or Author—That is the Question)

Lawrence M. Kendra
East Econ Data Systems

Jerry Clavner

Abstract

The purpose of this project is to compare the efficiency of three programming languages (BASIC, Pascal, and C) with that of a typical Authoring System (IBM Storey Board).

The vehicle for this comparison will be a simple interactive classroom presentation from the field of macroeconomics. The lesson has been flowcharted and then coded in all three programming languages. The lesson was also storey boarded and entered into the authoring system.

Comparisons:

1. Complexity of coding vs. authoring entry.
2. Investment required to purchase software.
3. Time required to learn software.
4. Ease of use in classroom.

This presentation will include a discussion of the effectiveness of the computer on classroom presentation and preparation time by the teacher, and a comparison of the design of the lesson using several different development tools.

Redesigning the Computer Science Curriculum in a Liberal Arts Setting

George T. Crocker
Susan T. Dean
Samford University

Abstract

Several factors motivate significant revision of our computer science curricula. Externally, there are the recommendations of the ACM/IEEE Joint Curriculum Task Force. Internally, we have both a need to increase the number of students in our programs and the opportunity to make a major contribution to a university-wide academic enhancement effort in honor of our sesquicentennial celebration.

Our goals are to

(1) design a flexible curriculum, based on the knowledge units set forth in the task force report, which encourages interdisciplinary degree programs and supports timely response to future externally motivated changes;

(2) build assessment into the curriculum;

(3) provide an experientially oriented computer science education that culminates in a capstone experience;

(4) incorporate recommendations arising from a survey of needs perceived by our alumni.

To support our new programs, we have proposed a dedicated computer lab facility for upper-level computer science students and faculty research. By providing more lab experiences and opportunities for the students to be guided into teaching themselves, we will produce graduates who are more independent and more attuned to the real-world environment.

Faculty Expectations of Computer use in Off-Campus Courses

John E. Cook
SUNY Institute of Technology at Utica/Rome

Abstract

Over 250 faculty have responded to a survey regarding computer use in the course(s) they taught off-campus in 1986-89. Such courses (e.g., military bases) are often part of professional programs. Judgement indicates that computer use should be common (e.g., business courses with an approximate 50% penetration). Only 28% of the faculty report student computer use was needed. Recent data from Fall 1989 (one-sixth of the total) shows an 11% increase in faculty computer use. For faculty wanting computer use, only 56% reported satisfaction with the arrangements. Also, 21% of faculty report that no appropriate library material was available.

Does limited access to computers or a library alter faculty perceptions of learning? Most faculty (92%) reported that their students learned as much or more than students in a comparable on-campus course. Tempering the response are numerous comments that learning was affected by adult student factors. The implication is that learning may be in spite of facilities, equipment, or services either available or not available.

The Role of Database Searching in Legal Education

Anne S. Caputo
The Catholic University of America

Rosalie M. Sanderson
University of Florida

Abstract

Database services such as LEXIS and WESTLAW have long been utilized in legal education as part of teaching the legal research process. This presentation contrasts the use of these legal information systems with that of DIALOG's family of nonlegal databases when second- and third-year law students at the University of Florida, College of Law enroll in advanced legal research seminars.

Specific project issues include: the role of nonlegal databases in supplementing and enhancing existing legal information systems; choosing the best method of database interface (command versus menu) for students to use; creating introductory material to inform students about the types of nonlegal databases available; preparing classroom presentations and exercises to acquaint students with a wide variety of available material and material formats (bibliographic versus nonbibliographic databases); and evaluating the student's choice of databases and patterns of usage with a view to expanding this pilot project to other law schools.

The Interactive Classroom

Elizabeth C. Brennan
Nova University

Abstract

This presentation will focus on the use of a distributed "co-op mini lab" mode of computer based learning experiences in an elementary school. The presentation will compare and contrast attributes of common methods of computer usage including labs, classroom-based computers, and distributed networks. An innovative model of instruction in which computers are integrated into the regular, daily, and ongoing curriculum will be discussed. There will be a brief overview of the objectives of a schoolwide project, research outcomes, and concepts associated with an "interactive classroom" model of instruction. A video prepared by actual students participating in the model will provide a detailed, close-up version of the theories, aspects, and learning activities experienced and associated with a mini-lab model. Of special interest is the facilitation of critical thinking skills and creative problem solving abilities as well as improved academic achievement through the use of small, seminar group instruction made easy with the use of computer "clusters."

The Development and Integration of Hypermedia CAI into an Elementary Reading Program

Randall Boone and Kyle Higgins
University of Washington

Abstract

This project is developing and testing state-of-the-art microcomputer software designed to aid elementary students (both learning disabled and nonlearning disabled) in the development of successful reading skills in a regular classroom setting. The research focuses on two areas of current interest in computer education: hypermedia instructional materials and integration of computers in a classroom.

The software provides hypermedia reading selections designed as supplementary material for a basal reader series. The hypermedia reading material offers easily accessible, additional information about the text, along with decoding and comprehension strategies within the context and physical structure of the reading selection itself. The software consists of a series of hypermedia lessons based on selected lesson segments from each basal grade level textbook in the series.

Data from the first year were examined by treatment grouping (experimental vs. control) and instructional sequence grouping (intervention either before or after teacher-directed activity). While almost no difference was found between experimental and control groups at any of the four grade levels when comparing entire classes, there is evidence that the intervention was a significant educational help for low achieving students.

Students who received the intervention before their teacher directed reading activity on the same lesson generally had higher gains in reading than did those who received the same intervention following the teacher directed reading activity. This suggests that the hypermedia lessons are perhaps best utilized as advance organizers than as follow-up activities.

What Do Preservice Teachers Need To Know About Technology?

Dianne I. Novak
The University of Michigan

Abstract

The question of what preservice teachers should know about technology has received increasing attention in the past few years. Beginning in 1987, the Preservice Technology Project has been funded by the Michigan Department of Education to address this issue. The Project was charged with developing policy recommendations concerning preservice technology education, and providing implementation strategies for teacher preparation institutions. To accomplish its goals, the Project established the Preservice Technology Training Task Force, a group consisting of a representative from each of the state's teacher preparation institutions. Facilitated by the Project, the Task Force has made suggestions for changes in the rules governing teacher preparation programs in the state, issued a report that makes comprehensive recommendations for future preservice technology education, and developed a handbook to assist teacher educators in the planning and implementation of technology at teacher preparation institutions. This session will summarize these past efforts of the Preservice Technology Project, and report on current and future activities.

Digital Systems Laboratory Course

Keith Barker
University of Connecticut

Abstract

Motivating sophomore-year students in their first real engineering course is one of the main topics of this tutorial presentation. One of the other main objectives of this course is to help students learn. This may sound strange when many other courses seem to have, as their objectives, assessment, evaluation, and grade discrimination. But in this course, we learn.

We learn about design, we learn about implementation and testing, and, most importantly, we learn about troubleshooting. And we do this by learning from and with each other. If the lab is quiet then there is something wrong; we are not sharing in the learning process. If the teaching assistant is standing by him- or herself then he or she is not being part of the learning.

Communication is also vital. We are educating engineers for a life-long process of learning and much of that learning takes place through professional communication. We learn together as the instructor communicates with the student and we use the students' individual communication as part of the assessment process. We encourage students to be good at written and oral communication as this is an important part of their professionism.

It seems to us that troubleshooting or fault finding is where we really learn. It is relatively easy to design at this level, to build and test, but in making mistakes in the implementation and testing we learn. We learn to trace forward and backtrace to find faults; we learn to think logically and how to fault find effectively and efficiently.

We can do most of this with the minimum of equipment and cost. It is nice to embellish the basic understanding with some sophistication, but that is not necessary. We use standard protoboards and components, cheap or free computer application programs, and the minimum of tools.

In doing this type of course we can provide many of the freedoms that we appreciate when learning. We don't have to do our lab homework only in the lab. We don't have to anguish over a faulty circuit only when the lab is open, we can learn all night if necessary.

This is a living course, it does not remain static, it grows and develops as feedback from the students and outsiders is received. It is being emulated by others because, not only do we learn well together, we have fun in doing so.

Hypermedia: Making the Learning Accountable

Ed Coughlin
Lake County Educational Service Center
Grayslake, Illinois

Abstract

Believing that hypermedia is one of the learning mediums of the future, we set out to experiment with its implementation in the classroom with these beliefs:

1. Hypermedia could more accurately deliver learning in methods compatible with the brain research and with the learning modality of a student.
2. Teachers directly involved with instruction could learn to create and tailor their own hypermedia sessions.
3. The hypermedia learning could be made accountable without compromising the flexibility of the medium.
4. A staff development model in the training of teachers to use hypermedia would be a necessary component of future teacher training.

Understanding the Importance of Computers in American Society

Harriet Copel and Carol Bryne
Lawrence Public Schools
Lawrence, New York

Abstract

In an effort to make students aware of the significance of computers in American society, an interdisciplinary curriculum project is used. Students use the computer as a tool while exploring its impact in our society. Initially, the district computer coordinator visited each class to show them the inside of the Apple IIe and gs computers, to discuss how information travels among the CPU and peripheral devices, to identify places of origin of the chips, and to discuss their experiences with computers outside school.

As part of their English curriculum, the students, in groups of two, created a survey that they administered with an interview to between 7 and 10 family members and local residents. The survey results were tabulated with a student-created *AppleWorks* database. As a group, students explored various ways of sorting the information gathered and after printing several reports, they wrote an interpretation of the data. As a group, the responses to the interview questions were analyzed and included as part of the interpretation.

Each student wrote a two-page research paper, using three articles from three periodicals, discussing computer use in America. A requirement was that the paper be written with a word processor. Students wrote papers about computer education pro-con, computer games, computer mapping, computerized music, computer viruses, types of computer and their uses.

Teaching Multiple Subjects using Technology

Frances Coleman
Choctaw County Schools
Ackerman, Mississippi

Abstract

Technology can allow students in small, rural schools to take courses that would otherwise be unavailable due to lack of money or personnel. This program has ben set up, not as a gifted program, but as a program open to any student who wished to take the courses in an effort to prepare himself/herself for college work and to compete on an equal footing with students from other environments. The students are scheduled into the course when they have available time in order to maximize student participation. This means that there are usually more than one and as many as six or seven courses being taught in the room at one time. Courses presently being offered through CAI include French 1, 2, 3, German, 1, 2, Latin 1, physics, AP physics, calculus, computer literacy, and beginning BASIC. Other courses are now offered by satellite as a part of this project.

Sound Waves: An Interdisciplinary Process Approach to Science

Denis M. Coffey and Janice C. Kowalczyk
Teacher Education and Computer Center

Abstract

Sound Waves is the first unit in a planned physical science program that demonstrates an interdisciplinary process approach to the learning and teaching of science in grades three to six. This unit integrates the computer with language arts, math, music, and art all in the support of the investigation of sound.

Children learn best when they are in control of their own learning and the material to be learned is relevant to their experience. Science can become more meaningful if it becomes part of a whole learning experience and is not treated as a separate subject. Teaching is a creative endeavor, therefore Sound Waves is designed to be a resource package.

LogoWriter is chosen as the medium of expression on the computer. Using the "sprites" and the built-in word processor, the students can extend their ideas, then play and construct in the areas of math, science, social science, writing, art, music, etc. Meaningful projects are created by combining all or some of these disciplines.

Sound Waves is being developed as a result of the Workforce 2000, "R.I. School of the Future" Project, and is being piloted by seven participating schools in the Spring of 1990. The presentation will include an overview of the Sound Waves unit, a demonstration of some classroom activities, and a glance at students' work.

Modeling: Tools for Learning Math and Science

Ricardo Nemirovsky, Chair
TERC
Cambridge, MA

Abstract

TERC has developed several projects for studying the potential of modeling activities in the learning of math and science. Currently a research project is being conducted at TERC on the basis of a grant from NSF. The project aims to understand how concepts of calculus can be learned through the modeling of real situations that students are able to observe and manipulate. This session will report partial results of this research. Other related projects will be reviewed. Several hands-on examples will be developed involving modeling software and computer tools for measuring and representing physical variables (MBL).

Participants

Robert Tinker
William (Tim) Barclay
Jan Mokros

TERC
Cambridge, MA

Sponsor: SCS

Women in Computer Science: Are We Failing in Our Efforts to Recruit and Retain Them in the Field?

Cindy Meyer Hanchey, Chair
Oklahoma Baptist University
Shawnee, OK

Abstract

Women in math and the sciences are becoming fewer and fewer. Or are they? This panel will examine an historical perspective of how and why women succeeded in science, will present a view of current trends of computer science students, and will offer some guidelines and concrete examples of what is being done and what can be done to encourage female participation in this area.

Panelists

Virginia Eaton
Northeast Louisiana University
Monroe, LA

Sharon Bell
New Orleans Public School System
New Orleans, LA

Susie Gallagher
University of Texas at Austin
Austin, TX

Lynda Nichol
Grove School
Shawnee, OK

Sponsor: SIGCSE

Sports Injuries

Kenneth E. Wright
The University of Alabama

Abstract

Sports Injuries, a computer assisted instructional (CAI) program, is an easy-to-follow, step-by-step computer program for the secondary and collegiate student athletic trainers. It is designed as a practical tool and will serve these three basic functions:

A. To assist the student athletic trainer in acquiring more detailed knowledge in the field of athletic training.

B. To permit the instructor to integrate information and practical application of athletic training techniques.

C. To allow the learner to work at his/her own pace.

This CAI consists of four units; ankle, knee, shoulder, elbow/wrist/hand. The organizing elements of each unit include: anatomy, basic treatment, common injuries, and injury evaluation format. At the conclusion of each unit, the athletic trainer will be asked to respond to various questions related to the proper steps in the evaluation and treatment of sports injuries. Through this learning process, the athletic trainer will become more familiar with their mistakes in evaluating and treating sport-related injuries.

Presently, Cramer Products has purchased and is marketing this computer assisted instructional program.

Research and Development Work of the Teacher Education Simulation Training (TEST) Group

Jerry W. Willis and Dee Anna Willis
East Carolina University

Abstract

Even a cursory look through catalogs of instructional software for colleges and universities will make the point that very little teacher education software is available. A few pioneers such as Powell [Powell, J. (1990) *Computerized simulation in the preparation of preservice teachers: A relational investigation.* Paper presented at the 1990 Association of Teacher Educators 70th Annual Meeting, Las Vegas.], Strang [Strang, H. (1990) *A self-administered simulation for training basic classroom skills.* Computers in the Schools, in press.], and Stroot, Tannehill, and O'Sullivan [Stroot, S., Tannehille, D., and O'Sullivan, M. (1990) *Skill analysis utilizing interactive video technology.* Computers in the Schools, in press.] have developed usable instructional software, but they are exceptions. Few teacher educators today are developing instructional software.

That is not the case in many other disciplines. A number of professors in economics, psychology, chemistry, medicine, nursing, mathematics, and history, for example, have developed programs that are now listed in the catalogs of distributors like Kinkos, Wisc-Ware, and Duke University Press.

In spite of the low level of activity, there are aspects of instruction at all levels of teacher education that can be improved through the use effective, well developed educational software. With that in mind the School of Education at East Carolina University, with grant support from IBM, is engaged in a program of technology diffusion that provides a wide range of support, including release time, graduate student support, consultation, instructional development assistance, training, and research support to faculty who wish to teach about or teach with technology.

One aspect of ECU's work is the Teacher Education Simulation Training (TEST) Group. The primary goal of the TEST Group at ECU is to develop and evaluate computer simulations for teacher education. We have targeted simulations that: (1) deal with widely taught topics, and (2) can be used independently by teacher education students in a computer laboratory. This presentation will briefly discuss the instructional design approach we use and the results of an experimental evaluation (using randomly assigned experimental and control groups) of the group's first simulation, IRIS. IRIS teaches students to administer and score an informal reading inventory. The presentation will also discuss ongoing work on additional teacher education simulations (e.g., I2C2–Integrating Instructional Computing into the Classroom), and will outline the approaches used at ECU to integrate the simulations into the teacher education curriculum.

Developmental Software Evaluation for Young Children

Susan W. Haugland
Southeast Missouri State University

Daniel D. Shade
University of Delaware

Abstract

In 1985, Drs. Haugland and Shade developed standards for the evaluation and development of software that matches the developmental characteristics of young children aged 3 to 8. Since that time they have evaluated and pilot tested nearly 200 pieces of software with young children. Their 10-point evaluation scale, first published in 1987 and recently revised, is congruent with the guidelines established for developmentally appropriate practice by the National Association for the Education of Young Children.

Conference participants will be introduced to each of the guidelines (age appropriateness, child in control, process orientation, independence, etc.) during a demonstration of the scale. The presenters will further assess the state-of-the-art of early childhood software and outline their conclusions concerning where software development is going and where it ought to go to impact early childhood education. Data from their research using the evaluation scale to select software and other sources of additional information will be made available.

The MSU Computer Competency Project

John Vinsonhaler, Hans Lee, and Leighton Price
Michigan State University

Chris Wagner
Oakland University

Abstract

The computer competency project (CCP) is an initiative of faculty at Michigan State and Oakland Universities. In the CCP, an interdisciplinary approach emphasizing cognitive science research is brought to bear on the problem of undergraduate computer education.

Instructional Approach: Computer Competency

The central thesis of the CCP is that traditional computer literacy does not meet the needs of the modern college student and should be replaced by computer competency. Computer competency stresses making people *users* of computers in their professions and in work places increasingly dominated by personal computers. This approach emphasizes productivity software applications in content disciplines, depth of content to the level needed in the real world, and learning how to learn about hardware and software. The CCP method is well represented by the textbook used [Vinsonhaler, Wagner, & Gentry. *People and Computers.* NY: West Publishing, 1989]. Techniques of instruction, based on cognitive schema theory, are built into the CCP curriculum and the textbook in the form of cognitive models, frames, scripts, and story schema.

The CCP Research Initiative: Measuring Computer Competency

The current research objectives of the project are:

1. Developing acceptable definitions of computer competency.
2. Developing tests of relevant declarative knowledge using multiple choice test items.
3. Developing tests of relevant procedural knowledge using performance items that measure the ability to use PC hardware and software.
4. Continuing the development of Socratech, a Prolog AI knowledge base capable of representing the above knowledge and aiding instructors with lecture and test preparation.
5. Contacting other researchers and teachers interested in joint research on computer competency.

Computer Coordinators—Do We Have All the Answers?

Rebecca A. Gold
Lawrence Township Schools
Lawrenceville, New Jersey

Elene A. Van Noy
Hopewell Valley Regional Schools
Pennington, New Jersey

Abstract

This presentation will provide a practical and effective way to attempt to handle all the day to day, month to month, and year to year planning and facilitating we are asked to do in our positions as computer/technology coordinators. We will explore the resources available to us to assist in our ever-expanding and demanding jobs, and the problems of keeping up with the newest technologies and their uses in education. From facilitating a district program to building new facilities and budget planning, we will share success stories, as well as things that did not work...and discuss why. We will discuss practical and proven techniques for working successfully with Boards of Education and the Administration, as well as building principals and professional staff. Applications of educational technologies across the curriculum, as well as teacher training and community involvement, will also be discussed.

This will be a presentation followed by an open group discussion and sharing session.

How to Plan, Prepare, and Present a Three-day, Multi-level, Hands-on Computer Workshop

John L. Tenny
Willamette University

Abstract

A description of a complex, three-day computer workshop that served from beginner to advanced users on both IBM and Macintosh computers, and resulted in increased awareness, experience, and enthusiasm will be discussed. Included is how to target surveys for greater response, use a database to identify talent and provide for direct needs of participants, and how to find and treat the needed computers. A description of the pitfalls, problems, and applications of Murphy's Law will be shared, along with copies of all surveys, registration forms, and schedules used in the workshop.

A Computer-Infused Teacher Education Program

Kathleen Maury
University of Wisconsin–Superior

Abstract

The University of Wisconsin–Superior has developed a microcomputer infused preservice teacher education program. While completing the professional coursework leading to teacher certification, undergraduate students use computers as professional tools, learn to use appropriate software to teach traditional school content, and learn to use and teach problem solving using the Logo programming environment. This is being accomplished with the Future Teachers Association, an open area resource center that houses print and nonprint curricular items and microcomputers, and a two-credit hour Problem Solving with Logo course. Students complete the program able to use technology as professional tools, select, use and teach with appropriate content related software, and the knowledge required to implement a problem solving curriculum using the Logo environment.

Using Computer Simulations as Part of a Field Based Model for Training Student Teacher Supervisors

Jane H. McHaney
Kennesaw State College

Abstract

This presentation will reflect the changing conditions in the world of professional field experiences. We have witnessed a growing number of reports criticizing the effectiveness of teachers and advocating a variety of reforms that would ostensibly improve the condition of education. The furor has produced, among other things, a more visible emphasis on accountability. It can be observed that the effective teaching movement is a response for the demand for accountability. The result has been a number of criteria developed through observation and research that can be described and observed.

In this program the individual's progress is determined by demonstrated competency. Competencies are derived from explicit conceptions of teacher roles, and the individuals performance is assessed in relation to specific, previously identified competencies. Information relating to each of the objectives will be shared during this presentation. An opportunity to participate in the computer simulations will also be included in the session. The simulations include experiences for the university supervisor, student teacher, and supervising teacher.

41

Fitting the Pieces Together: Successful Technology Implementation

Hilary Cowan
Kanawha County Schools
Charleston, West Virginia

Abstract

Successful technology implementation for schools and districts is not a matter of luck or happenstance. Although enthusiastic individuals can initiate such an implementation, the total program should not be allowed to rest on the assurance of their ongoing participation and commitment to the program. This bottom-up approach to implementation suffers from an inconsistency of training and from a reliance on people who may suffer burnout from being overworked and underpaid for their expertise. On the other hand, top-down implementation may be resented by teachers and staff who have not participated in the initial planning and who feel harassed by the constant onslaught of new ideas and programs that they are obligated to implement into their curriculum plans.

Successful implementation is dependent on effecting a behavioral change on the part of the teachers involved in the training. Research shows that such change takes a great deal of time, support, training, and enthusiasm from all participants. These variables must be locked in place with long-term planning, consistent focus, synthesized implementation, and community support.

In 1987, Kanawha County Schools in Charleston, West Virginia, began such a plan for over 1,200 elementary school teachers in 74 schools. The components of this successful plan will be defined. The unexpected "political" aspects of planning, which can interfere with keeping the plan on track, will also be discussed.

Project Get Ahead

C. A. Coulter
SER-Jobs for Progress, Inc.

Abstract

Project Get Ahead is an educational project that promotes high school completion for in-school youth. The project also serves out-of-school youth by providing GED preparation and employment. The project serves 150 low-income youth between the ages of 14 and 22 in the Santa Fe, New Mexico area. Project participants include at-risk high school students, high school drop-outs, and teen parents. The program also serves young people in residential treatment homes.

Individualized basic skills instruction is provided to each student by computer assisted instruction at one of three computer labs using a UNISYS ICON network. Each computer lab site delivers the Autoskills Reading, the Autoskills Math, and English One programs. Students spend an average of five hours each week on basic skill remediation. Training in pre-employment skills, such as completing job applications and preparing for job interviews, is also provided. Students are assessed both pre and post by means of the TABE. Preliminary results indicate an improvement in basic skill achievement levels and a reduction in the high school drop-out rate for project participants.

42

Computer Generated Biofeedback for Emotionally Disturbed High School Students

Ned Davis
Pequannock Township High School
Pompton Plains, New Jersey

Abstract

Students in an alternative high school for emotionally disturbed juniors and seniors have been using computers to help them learn the normal function of their bodies and how emotions effect their physiological balance. This project is concerned with:

- Students learning basic physiology of the heart and brain as part of their unit in biology.
- Students learning how the body reacts to various emotional states by monitoring their own heart, brain, and muscular tension.
- Students learning to reduce high stress levels using coping techniques.

Sensors attached to areas of their legs, heart, and head receive impulses that are converted into polygraph readings onto a computer monitor and also saved to disk. They "see" how different emotional states effect their heart rate and brain activities. With careful direction by the teacher and school psychologist, these ED students learn coping techniques using computerized biofeedback.

This program has fascination implications for developing learning readiness in public education.

The BUDDYSYSTEM Computer Project: The Curriculum-Computer Connection

Linda M. Curry and Sharon Snellenberger
Indianapolis Public Schools

Abstract

Washington Irving Elementary, Indianapolis Public Schools, is one of five sites in Indiana involved in the BUDDYSYSTEM Take-Home Project. Each of the students in the fourth and fifth grades have an Apple IIgs personal computer at home. Hopefully, the computers will stay in the students' homes through the sixth grade for student, parent, and sibling use

The presentation of our project will be about our involvement in the BUDDYSYSTEM and the implementation of the computer as a tool both at home and in the classroom. The oral presentation will be illustrated using a slide show created on Scholastic's "Slide Shop." A packet of multidiscliplinary lesson plans, which have been used to integrate the computer as a tool into the curriculum , will be handed out to participitants.

Employing the Disabled: New Legislation, New Programs, New Technology!

Rita Thomas Noel
Western Carolina University
Cullowhee, NC 28723
704-227-7401

Abstract

This paper examines the new legislation affecting the employment of disabled American and the impact of computer-related assistive technologies. The facts which have contributed to the current labor situation are discussed: the changing workplace, the changing nature of work, and the changing workforce.

preparing the disabled for employment means helping the individual adapt to job limitations through job restructuring and assistive technology. Programs provided by State Research and Training Centers are discussed as sources of information for developing adaptive devices. Low Technology and High Technology devices are compared and examples provided of new assistive technologies in use with specific disabilities. A cost-per-day system is recommended when examining the price of more expensive technologies.

A selected list of centralized information clearinghouses is also presented. Rehabilitation/Industry Partnerships are discussed as well as Secondary Education Transition Programs using assistive computer-related technologies. The 15 year results of IBM's computer programming for the severely disabled is also provided. Included with the report is a current bibliography.

More than 43,000,000 Americans are now recognized as disabled. Encouraged to seek employment by the 1989 Americans with Disabilities Act (ADA) disabled youth and adults will need the help of well-trained educators to find employment in a changing workplace. The technology is available to help teach a wide variety of job skills to a diverse population of learners. Training can now be offered through telecommunications, interactive video, computer-assisted instruction, and through an array of simulated, computerized training devices. However, it may be the assistive role of technology that provides the greatest source of empowerment for the disabled who seek employment. What has prompted the growing concern for providing employment opportunities for the disabled? Examining the factors contributing to the new meaning of work helps to explain the employment dilemma.

The Changing Workplace

Computers, FAX machines, and programmable equipment are changing *where*, *when* and *how* we work. Less than 3 percent of the U.S. population is employed in food production, yet surpluses of agricultural products are exported. Manufacturing, too, has experienced a loss of workers. Today, less than 30 percent of the workforce is needed to produce manufactured goods. Factory jobs have become highly automated processes. Some labor economists predict that this trend will continue as a result of developments in automation and computer technologies.

Over half of the workforce today are white collar workers. For the first time in human history, jobs that depend on brawn are outnumbered by those that don't. Even in the service industries, dependence on technology is rapidly increasing. Computer generated voice messages and large voice-data networks now perform many services electronically and quite automatically. The performance of all office work (and who is producing that work), has been radically changed by the use of software applications and computerized equipment. The increased use of electronic processing (telephone lines linking home computers to office terminals) has made office work independent of geographic location. The proliferation of home industries utilizing emerging technologies has sharply increased in the past five years.

Changing too, is how we perform those traditional office tasks. Work stations, which include a computer terminal with networking capacity, a FAX machine, printer, and often an optical scanner allows an office worker to perform a wide range of functions without ever leaving his/her desk. Paper is not transferred from one place to another; all the necessary forms and records are available on the screen.

Technological improvements have increased and maintained high productivity in the workplace. However, these advancements reduce the jobs available for blue collar workers and some kinds of white collar work creating a highly competitive job market.

A Changing Work Force

America's work force is changing, too. The number of traditional applicants for job openings is decreasing. Labor economists are predicting a continued slowdown in the availability of young workers. The 1988 Populations Projections (U.S. Bureau of the Census) forecasts a shrinkage of 2,000,000 18-24 year-olds in the available labor market by the year 1995. Businesses in more affluent suburbs have already indicated a lack of available teen-age hires.

In contrast, the number of older Americans has steadily increased. By the year 2,000 we will have as many people over the age of 75 as we have under the age of 5. The increased life-span for average Americans means an increasing number of older workers who may desire part-time work or in some cases, second careers. A higher retiree-to-worker ratio creates a greater dependency of the non-working on the shrinking numbers of those who are working.

Another segment of the new workforce is the thousands of workers diagnosed as AIDs victims or suffering from HIV infections. The explosive growth in numbers of these disabled workers creates expanding unemployment.

These changes indicate to employers, educators, and to those involved in the training of employees the need to reevaluate how and who is vocationally trained. To provide a productive workforce, which allows all of us to enjoy quality lives, America cannot consider any of her citizens as potential "Throw-aways."

Why Hire the Disabled?

Meaningful employment provides the handicapped individual with an enhanced quality of life, with the accompanying self-respect of independence and personal autonomy, and wit the opportunity to participate in the life of a community. The value of the contribution of the disabled person to the workplace and to the lives of other employees is seldom included in the cost/benefit ratios.

For those who are not employed, lifetime disability benefits payments plus the cost of medical and social services received may easily amount to more than $1,000,000. The Federal government regards the value of a job to public revenues to equal $65,000 in total taxes paid, in the saved costs of social services, and in the ripple effect from consumer spending. To those empowered by employment, however, attaching only a dollar value to a job based on an income-generated versus services-received formula represents a limited perception of employment.

Learning to Match Job Limitations with Assistive Devices

The Americans with Disabilities Act (ADA) and the Technology-Related Assistance Act of 1988 are strengthened by additional legislation provided in the ADA of 1989. Legislation now supports and funds adaptive technologies, architectural access, and provides for occupational training to the disabled. Disabled Americans are assured of non-discriminatory employment practices, but how can they be assured of employment opportunity?

Determining the extent of employment opportunity for the disabled can be accomplished using a two step approach. Originally developed in 1986 by the University of Wisconsin Stout as a "rehabilitation technology schema," it works equally well with re-training an injured or newly disabled employee or when training an employee with a known disability. It consists of:

1. Rethinking the job description, by using compensatory strategies--the restructuring of tasks, activities and the environment to compensate for the functional limitation experienced by an individual, and
2. Identifying possible adaptation of assistive devices and equipment.

The challenge is to know which devices are available, which are the most appropriate, and how to acquire the needed information and training.

The National Association of Rehabilitation Research and Training Centers (NARRTC)

One source of information is the directory of research and training centers, listed by state, available from the National Rehabilitation Information Center. The directory provides a comprehensive listing of the relevant research, training, and service demonstrations conducted throughout the United States. Twenty-three states are represented by one or more programs which use computer technology in testing and developing assistive devices or in teaching/training the disabled.

Descriptions of the programs identify special populations, physically disabled, hearing impaired and deaf, brain injury, blind and visually impaired, and rural area programs. Many of the programs address the problems of infancy through adulthood as well as research, community organizations, family life, and employment practices.

Assistive Technology and Equipment

How can designers of training for the disabled match existing needs with existing equipment? Accommodation devices can be simplistically categorized as "low technology" devices and "high technology" devices. Low technology devices and aids are both simple in design and low in cost. Low technology devices have other user benefits, too, because they do not require costly maintenance and seldom require user training. A selected comparison in Table 1 illustrates the wide range of technology included in dramatic state-of-the-art accommodations as compared to those representing simple general improvements in worksite design or environment. Over 2,000 manufacturing firms now make aids for the handicapped (according to Social Issues and Health Review (SIRH) editor Mark Fadiman in *American Demographics*, August 1987, p. 20) and, more than 800 products are now being marketed to help physically disabled computer users.

For Users With Limited Mobility. Reflector technology uses infrared beams to prompt an optical camera monitor (located in a small black box on top of

Low Technology Devices	**High Technology Devices**
velcro for fasteners	*PRAB robotic devices*
walking cane	*puff-sip control wheelchairs*
door levers instead of knobs	*portable communication aids with speech synthesizers*
lapboards	*screen reading devices*
headset phones	*optical character scanners for inputting data without keyboard or voice*
arranging for delivered lunches	*portable respirators*
changing desk layout from right to left	*keyboard emulators*
telephone amplifier	*telecommunications devices for the Deaf and Dual Party Relay Systems*
automatic page turner	
providing training in disability to supervisors and other employees	
plexiglass keyguard for computer	
computer screen magnifier	

Table 1. A Comparison of Technical Complexities of Assistive Devices for the Disabled.

the microcomputer) to control the computer's cursor. The reflector, about a inch in diameter, is a pointing device which can be attached to eyeglasses, hat, or head-band of the user. A nod of the head then moves the prompt control. Pointer Systems has also introduced the FreeWheel system, another wireless pointing device that moves the microcomputer cursor for the handicapped user (*PC Week*, July 4, 1988).

Personics markets the HeadMaster headset for MacIntosh users who are paralyzed. Users can input data by moving the cursor to the appropriate key before puffing air into a mouth tube connected to the headset. A character is formed on the screen by each puff.

Even the simplification of key-strokes to allow the user to type single-handedly by combining two key functions into one can provide the disabled user new access to the computer keyboard. Easy Access, another MacIntosh program, changes the numerical keypad into cursor controls. Both adaptive programs are part of the systems tools and are available without additional charge to users (*MacUser*, February, 1988, p. 256).

Eye-Typer is another product which allows a paralyzed user to focus on the screen where a depiction of a keyboard is shown. A camera senses the eye movement and the eyed letter is displayed on the screen. It is not particularly a fast system, but is enables the severely disabled access to a direct messaging system (*Fontana*, October 10, 1989).

For the Blind and The Visually Impaired. Users who are accustomed to the braille system, can use VersaBraille II + a microcomputer display that uses raised figures along a 20 character panel. Versapoint is another option providing a braille embosser with some graphics capability. Braille users can also use a dedicated PC with the aid of an optical scanner and a voice synthesizer. A braille printer can then be attached for output. However, since only 1 in 8 of the visually impaired read braille, the voice synthesizers and screen reading software applications can be used with a regular computer keyboard. When a scanner is used as the input device, it scans a page of information and inputs the document through a decimated word processor. The sightless operator then scans the document for verification through speech or braille output. The new system is faster and easier to use than a reading machine and usually much less expensive.

For the Deaf and Hearing Impaired. IBM's SpeechViewer allows deaf users an opportunity to participate in a hearing world. The computer produces a vivid graphical indicator to represent the sounds made by the user. These images provide voice mapping by creating a colored oscilloscope-like-diagram responding to the users voice. SpeechViewer provides a way for the hearing impaired user to practice and recognize correct speaking patterns without suffering the embarrassment of public trail-and-error.

Cost of Equipment

Even when considering devices as "high technology," dividing the initial purchase price into a cost-per-day may more than justify their use while still representing a cost-effective investment. For example, a PC equipped with screen reading software and a voice synthesizer which would allow a visually impaired user to hear what was being typed, may cost approximately $2,500 for the total work station (top-of-the-line products could cost considerable more). A non-adapted PC varies from $1,200--$2,000 leaving $1,300 as the investment in adaptive devices. Amortizing the cost over a one-year period, investment for equipment represents $5.20 per working day. Extending the investment figure over two or more years makes the daily cost of equipment almost negligible.

Where to Find Information

The President's Committee on Employment on People with Disabilities and private organizations fund specialized information centers free of charge to employers, parents, educators, and the disabled worker. Six examples of those organizations providing information on adaptive technologies are:

JAN Job Accommodation Network
•1-800-526-7234
•8 a.m.--8 p.m. EST
An information network and consulting service providing individualized accommodation solutions to employers.

ABLEDATA Newington Children's Hospital
1-800-344-5405
Provides a list of commercially available products for a variety of applications, including job accommodations.

AT&T National Special Needs Center
1-800-233-1222
8:30 a.m.--7:00 p.m. EST
Provides services to order equipment necessary for job accommodates for the hearing-impaired and deaf.

Rural Research and Training Center
405-243-5481
Provides information on assistive technology for people in rural occupations such as farm/ranch workers who have experienced a physical disability.

The National Support Center
1-800-IBM-2133
Mary Pat Radabaugh, Director
Employers, healthcare professionals, and consumers can see demonstrations of new products and software, as well as adaptive technologies. A comprehensive listing of companies and available adaptive devices is also available.

National Rehabilitation Information Center (NARIC)
1-800-346-2742
Maintains a clearinghouse on services and equipment available.

Rehabilitation/Industry Partnerships

Recently a number of new relationships have been established between the rehabilitation community and industry. In addition to providing training, industry is attempting to improve the availability of accessible technology. The Electronics Industry Association has formed an Assistive Devices Division (ADD) which is made up of manufacturers of special needs electronic products, computers, and telecommunications products. Hopefully, this will result in the development and the delivery of accessible technologies to all people with disabilities. IBM, the National Easter Seal Society, and United Cerebral Palsy offer selected PS/2 personal computer products at a discount to qualified individuals with disabilities.

Apple Computer has formed a grassroots network of rehabilitation centers and disability organizations called the National Special Education Alliance. Its purpose, too, is to share information about accessible technology and consumers' disability needs.

AT&T, leaders in providing information to employers and consumers in the field of adaptive audio devices, have joined with the Robert Wood Johnson Rehabilitation Institute to form the Easy Living Laboratory. This laboratory serves as a demonstration and training facility for rehabilitation professionals and consumers with disabilities.

Secondary Education and Transition Services

Services and intervention are available for disabled individuals of all ages, but the Office of Special Education and Rehabilitation Services (OSERS) has focused on the problems of handicapped youth. Yearly grants and contracts, totally more than $6.6 million, are available to assist in transition training and education for handicapped youth exiting the secondary school. Most of these programs are using adaptive technologies for the physically disabled and are using some form of computer-assisted instruction in training youth for employment.

The Secondary Transition Intervention Effectiveness Institute's (Transitions Institute) 1988 *Compendium* describes the status of 180 demonstration projects funded by the U.S. Department of Education, Office of Special Education and Rehabilitative Services.

Transitions estimates that to-date 127,790 youth with handicapping conditions have been impacted by the model demonstration projects and approximately 22,567 youth have received direct services.

Projects With Industry

As early as 1972, the federal government recognized the necessity to involve industry with job training and placement for the disabled. IBM's Systems Integration Division (SID) assists state rehabilitation agencies in developing and maintaining computer programmer training and placement programs for severely physically disabled persons. At present, there are 33 operating training projects in the United States. In 1988, the 33 programmer training projects graduated 284 students and successfully placed 215 with a total of 2,330 placements (85%) since the project began in 1973 (IBM Annual Report, 1988, section 3, page 4).

A Parting Thought

For most of us the use of technological advances and application of technology makes things easier. We will continue to look for ways to make our own lives easier and more efficient. For people with disabilities, technology often means the difference between dependence and independence. Those who are disabled will look to those of us who can to lead the way in providing opportunities for using and integrating technology into the classroom and into the work place. Technology for those who are disabled not only makes things easier, it can often make things possible.

REFERENCES

Brody, H. (July, 1989). Instant equalizer. *PC/Computing*, pp. 83-93.

Chandler, D. (May 24, 1988). PC revolution enlists the handicapped. *PC Week*, V5, p. 17.

Condeluci, A. (April/May/June, 1989). Empowering people with cerebral palsy. *Journal of Rehabilitation*, V54, 2, pp. 15-16.

Dowling, J. & Hartwell, C. (1988). *Compendium of Project Profiles 1988*, Secondary Transitions Intervention Effectiveness Institute, University of Illinois, Champaign, IL.

Edmondson, B. (August, 1987). Enabling the disabled. *American Demographics*, V9, p. 20.

Fontana, L. (October 11, 1989). Personal Telephone Interview, Visual-Impairment Specialist for JAN (Job Accommodation Network). 1-800-526-7234.

Hansen, C. E. & Perlman, L. G. (July/August/September, 1989). Technology: A vital tool for persons with disabilities. *Journal of Rehabilitation*, V55, *3*, pp. 18-21.

IBM. (1989). Computer programmer training for the severely physically disabled. Annual Report 1 January, 1988-31, December, 1988.

Mendelsohn, S. (1989). Policy issues in technology. (In) *Technology and Employment of Persons with Disabilities*. 13th Mary E. Switzer Monograph, National Rehabilitation Association, Alexandria: VA.

National Association of Rehabilitation Research and Training Centers (1988). *1988 Directory of Rehabilitation Research and Training Centers*. Office of Special Education and Rehabilitative Services. U. S. Department of Education, Washington D.C. 20202

United States Congress (1989). PL-933 Americans With Disabilities Act of 1989, pp. 1-26.

Valiulis, D. (February, 1988). Easy Access gives people with movement disabilities full access to Mac software. *MacUsers*, V4, p. 256.

Weed, R. & Whitescarver, C. (July/August/September, 1989). The exciting potential of hi-tech workstations. *Journal of Rehabilitation*, V55, *3*, pp. 10-12.

An Inductive Inference Approach to Plagiarism Detection in Computer Programs

Gerard K. Rambally
Mauricio Le Sage
Department of Computer Science
University of New Orleans
New Orleans, La. 70148
504-286-7361

Abstract

An automated plagiarism detection system which was developed and implemented in LISP on a VAX 8600 is described. The system has been tested on a number of programming assignments in an introductory Computer Science class in which PASCAL is used. The results to date are encouraging. The plagiarism detection system accepts as input all the students' programs for a particular assignment. It then parses each program to generate a knowledge system which contains knowledge vectors characterizing each program in terms of fourteen attributes. This knowledge system is then passed to the ID3 algorithm, which uses a precise measure of information called entropy to classify the programs in a decision tree. Using this decision tree, we are able to identify (a) programs which were not involved in plagiarism, (b) programs in which plagiarism is suspected, and (c) programs which were not graded fairly.

Introduction

Surveys have shown that the problem of program plagiarism in out-of-class assignments plagues the introductory programming courses offered by the Computer Science Departments of most universities in this country (Shaw, Jones, Kneuven, McDermett, Miller, Notken, 1980). Some factors which facilitate cheating in these classes are large enrollments in these courses, and automated facilities that allow files to be copied almost effortlessly.

From their surveys of Computer Science Departments, Shaw et al (1980) have compiled a number of suggestions about the prevention of plagiarism; however, regarding the detection of plagiarism, their studies have shown that most departments rely on astute graders to detect (almost) identical programs. However, the effectiveness of graders in detecting plagiarism is severely limited simply because a grader for a large class cannot remember all previously graded programs while examining the current one.

An alternative to humans detecting plagiarism is computerized methods of identifying program similarity. This is precisely the approach taken in this research project. Automated plagiarism detection systems do not rely on labor intensive methods (as does plagiarism detection by graders) and are, therefore, more cost-effective. A greater advantage however, is that automated plagiarism detection systems are more fair than plagiarism detection by graders since the method is more systematic and not as random as in the latter case.

In this paper we describe an automated system that identifies Pascal programs that are "almost" identical. This system was recently developed and implemented in LISP on a VAX 8600 at the University of New Orleans. It is important to stress that this system only identifies programs which are suspected of plagiarism; it is left to the instructor to pass final judgement.

A few automated plagiarism detection systems have been built. Most of these identify programs that are identical by comparing program attribute vectors $<a_1,a_2,\dots,a_n>$ where each a_i is a count of the occurrences of a particular attribute. A comparison routine is then invoked which detects similarities between vectors. Ottenstein (1976) based his system on Halstead parameters. The resulting program attribute vector consisted of a quadruple $<N_1,N_2,n_1,n_2>$ where N_1 = total operators, N_2 = total operands, n_1 = unique operators, and n_2 = unique operands. The four-tuples of programs are compared and programs with identical four-tuples are considered for further investigation. Grier's (1981) system assigns weights to each value in the vector according to the relevance of the corresponding attribute to the assignment. A compaction routine is then used to replace each vector with a single value. Donaldson, et al. (1981) used a similar method of weighting the values in the vector. In addition, they also summed the differences between corresponding attribute values for each vector pair, and maintained a count of the number of features for which the absolute difference was zero. Another plagiarism detection system has been incorporated in a program analysis system called Institutional Tool for Program ADvising (ITPAD) (Robinson & Soffa, 1980). This system builds a graph to represent the structure of each student's program. It then compares pairs of programs by counting attributes of this representation.

There is considerable similarity with regard to the attributes used in the above mentioned systems and our plagiarism detection system. However, many of these are compound attributes and some authors expanded some attributes into sub-attributes while others collapse several attributes into one. For example, both Donaldson (1981) and Grier (1981) count variants of loop-building statements separately, while in our system we collapse all such statements into one attribute. Our system differs significantly from the systems above with respect to the classification procedure used to cluster the programs. Our method applies a precise measure of information called entropy (discussed in section 3) to the classification problem.

The Knowledge Representation System

The basic component of the knowledge system is a set *P* of programs. Knowledge about these programs is expressed through a set *A* of attributes which characterize the programs. Thus, for each $a \in A$, a set V_a of attribute values will form the components of the knowledge system. Furthermore, we will assume the existence of a function *f* which assigns attribute values to the programs. This task is performed by the parser in the plagiarism detection system. Formally, the knowledge representation system is a quadruple

$$S = (P, A, V, f)$$

where P is a non-empty, finite set of programs,

A is a non-empty, finite set of attributes which characterize the programs,

$V = \cup_{a \in A} V_a$ is the non-empty, finite set of values of attributes, and

$f : P \times A \rightarrow V$ is a knowledge function such that $f(p,a) \in V_a$, $\forall p \in P$, $a \in A$.

The function $f_p : P \rightarrow V$ such that $f_p(a) = f(p,a)$ for every $p \in P$, $a \in A$ will be called knowledge about p in S.

This knowledge system may be considered as a finite table in which columns are labeled by attributes, rows are labelled by programs, and an entry in the p^{th} row and a^{th} column has the value $f(p,a)$. Each row in the table represents some knowledge about some program in S.

We will illustrate this formal definition of the knowledge system with a set of five programs:

P= $\{p_1, p_2, p_3, p_4, p_5\}$

A= {class (a_1), num_of_variables (a_2), assignment_statements (a_3), num_of_loops (a_4), num_of_decisions (a_5), num_of_subprog (a_6), total_operators (a_7), unique_operators (a_8), total_operands (a_9), unique_operands (a_{10}), routine_calls (a_{11}), num_of_arrays (a_{12}), num_of_records (a_{13}), num_of_pointers (a_{14})}

$$V_{\text{class}} = \{A, B, C, D, F\}$$
$$V_{\text{num\_of\_variables}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{assignment\_statements}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{num\_of\_loops}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{num\_of\_decisions}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{num\_of\_subprog}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{total\_operators}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{unique\_operators}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{total\_operands}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{unique\_operands}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{routine\_calls}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{num\_of\_arrays}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{num\_of\_records}} = \{0,1,2,\ldots,n\}$$
$$V_{\text{num\_of\_pointers}} = \{0,1,2,\ldots,n\}$$

f is defined by Table 1.

Note that these are the actual attributes and the actual domains of the values of these attributes used in the research project. Attribute class refers to the letter grade assigned by the grader to each program. *num_of_variables* refers to the number of variables which are declared and used. *assignment_statements* refers to the number of assignment statements minus the number of initialization statements. *num_of_loops* includes the number of **while**, **for**, and **repeat** loops, and the *number_of_decisions* refers to the number of **if** and **case** statements. *total_operators*, *unique_operators*, *total_operands*, and *unique_operands* are Halstead's metrics. The description of the other attributes are obvious. The decision to use lexical metrics (that is, ones which count occurrences of certain types of syntactic elements) exclusively, was based on the evidence that the programs under study are written by a homogeneous group with the same goal in mind, and syntax oriented metrics are likely the most effective for discriminating among programs that perform the same task.

Inductive Inference using ID3

	Attributes													
Programs	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}
p_1	B	42	14	10	16	7	38	9	34	23	7	8	0	0
p_2	A	46	14	11	13	7	29	9	25	18	7	8	0	0
p_3	F	48	19	11	13	7	32	9	28	18	7	11	0	0
p_4	C	41	15	11	4	6	19	8	19	1	6	7	0	0
p_5	F	49	20	11	11	7	32	9	28	19	7	11	0	0

Table 1. The knowledge system.

This section discusses the ID3 algorithm (Quinlan, 1983) which uses the process of inductive inference to identify a classification rule from the data in our knowledge system. The ID3 algorithm is the most commonly used algorithm in commercial expert systems that employ inductive methods to generate rules. *ID3*, as used in this project, takes programs of a known class (a letter grade of A, B, C, D, or F) described in terms of the fourteen attributes in our knowledge system, and produces a decision tree over these attributes, that correctly classifies the given programs.

A classification rule in the form of a decision tree can be constructed for any set P of programs. If all programs in P belong to the same class, then the decision tree is a leaf that bears that class name. Otherwise, P contains representatives of different classes; we select an attribute and partition P into disjoint sets $P_1, P_2, \ldots, P_n$, where P_i contains those members of P that have the i^{th} value of the selected attribute. This rule-forming strategy is performed on each subset of programs which results in ID3 partitioning the set of programs into equivalent classes. The eventual outcome is a tree in which leaves may have

a) a single element (program). In this case no other program in set P is equivalent to this one. If each of the leaves in the decision tree have but one element, no plagiarism has occurred.

b) more than one element (program), all with the same grade. In this case, the programs in this leaf are suspected of plagiarism and the instructor should investigate further the content of these programs. Note that we are not saying that programs with the same grade in the same leaf definitely implies that plagiarism occurred, since it is possible for equivalent programs to be written independently. This case should however, raise a flag for the instructor to investigate further.

c) more than one program, all of which are not in the same class. This case may indicate that there has been some inconsistency in grading the programs, and again the instructor should be alerted to investigate further for possible unfair grading practices.

To illustrate this classification process, consider the knowledge system in the previous section. Obviously the table alone does not give us much insight into deciding where plagiarism has occurred. What is needed is a way to use the examples in the table to produce a classification tree.

To build a classification tree, one of the attributes is selected to be the starting point or root node of the tree. Once this attribute is selected the set P is partitioned into disjoint subsets, each containing programs with the same value of the selected attribute. If $P \supseteq X$, and X contains programs of only a single class, then no further classification is required on set X. If $P \supseteq Y$, and Y contains programs of different classes, then further classification is needed, that is, another attribute must be selected and the set Y must be partitioned further into disjoint subsets, and continue in this fashion until each leaf in the classification tree has programs of the same class or until the attributes are exhausted.

The whole skill in this style of induction lies in selecting attributes that lead to efficient classification trees. ID3 uses an information-theoretic approach aimed at minimizing the expected number of tests to classify an object. Entropy is used to measure the amount of information about classification contained in a single attribute. As the entropy increases, the amount of information that we gain by knowledge of the final classification increases. In this application, an object can be classified into 5 different classes, A, B, C, D, or F, and if we let $p(x)$ represent the probability of a program being in class x, then the entropy of classification, $H(c)$ is

$$H(c) = -\sum_{i = A,B,C,D,F} p(i) \log_2 p(i)$$

with a known set P of programs we can approximate these probabilities by relative frequencies, so that $p(A)$ becomes the proportion of programs in P with class "*A*." The entropy of classification for the knowledge system above is

$$\begin{aligned} H(c) = & -(1/5)\log_2(1/5) - (1/5)\log_2(1/5) \\ & -(1/5)\log_2(1/5) - (0/5)\log_2(0/5) \\ & -(2/5)\log_2(2/5). \end{aligned}$$

$$H(c) = 1.921928$$

Now we need to decide which of the attributes in A to test next. The values V_i of attribute I are mutually exclusive, and the amount of information contained in attribute I will be

$$B(c|I) = (\textit{probability that value of } I \textit{ is } V_i) \times H(c_i)$$

where again we can replace the probabilities by relative frequencies. The attribute to test next is obviously the one which gains the most information, that is, for which

$$H(c) - B(c|I)$$

is maximal.

From Table 2, $H(c) - B(c|I)$ is maximal for *num_of_variables*, and ID3 will select this attribute to form the root of the decision tree. From the tree generated (Figure 1), branches "46" and "49" require no further work, while the branch "42" needs to be split further. After recomputing $H(c) - B(c|I)$, the attribute chosen is "*num_of_loops*" and the resulting tree is Figure 1.

B(c|I)	H(c) - B(c|I)
B(c|num_of_variables)	1.921928
B(c|assignment_statements)	1.521928
B(c|num_of_loops)	0.721928
B(c|num_of_decisions)	1.521928
B(c|num_of_subprog)	0.721928
B(c|total_operators)	1.921928
B(c|unique_operators)	0.721928
B(c|total_operands)	1.921928
B(c|unique_operands)	1.521928
B(c|routine_calls)	0.721928
B(c|num_of_arrays)	1.521928
B(c|num_of_records)	0.0
B(c|num_of_pointers)	0.0

Table 2. Information gained from each of the attributes examined.

The Plagiarism Detection System

To use the plagiarism detection system, the instructor first copies the programs for a particular assignment from the students' accounts into a file called HOMEWORK_X (where X represents the assignment number). The plagiarism system then uses this file, HOMEWORK_X, as an input file and parses each program to determine the values that should be assigned to the fourteen attributes in our knowledge system. Table 3 shows the knowledge system generated for an assignment in a class of fifteen students.

Because of the counting algorithms, most of the cosmetic changes on plagiarized programs will be transparent to this approach. Cosmetic changes such as insignificant permutations of statements, re-commenting, unneeded initializations and declarations, reformatting input/output statements, and renaming variables will have no effect on the values assigned to the attributes in the knowledge system. For example, the system keeps track of the variables declared and used. Hence the obvious tactic of changing variable names and/or excess declarations are an ineffective change to a program. Unneeded initializations, declarations, and input/output statements are taken care off by simply not counting them. Thus, reformatting input/output statements is also an ineffective change to a program.

The next step in the plagiarism detection process is to identify the attribute I for which $H(c)$ - $B(c|I)$ is maximal. In the example shown in Table 3, attribute *total_operands* gains the most information. The plagiarism system then modifies the knowledge system so that the values of the particular attribute chosen, which fall within predefined intervals of tolerance, are equalized. The predefined intervals of tolerance for the attributes are shown in Table 4.

This scheme was developed and tuned by using groups of programs from an introductory PASCAL class. This table says, for example, that if the number of assignment statements in one program differs from the number of assignment statements in another program by two or less, then the same number of statements is assigned to each program. The modified knowledge system is shown in Table 5.

This modified knowledge system is then used by ID3 to classify the programs. These last two steps are repeated until the complete classification tree is generated (see Figure 2).

Results and Discussion

As the classification tree in Figure 2 indicates, two leaves have two programs in each of them. Since the porgrams in each of these leaves are in the same class, these leaves contain the only possibilities of plagiarism. Note that many of the attributes were not considered by ID3 in developing this classification tree. Thus it is possible for two or more programs in the same class to be classified in the same leaf and yet be independently produced. Thus, before the instructor is notified as to whether to investigate programs for possible plagiarism, the entire attribute vectors of programs in the same

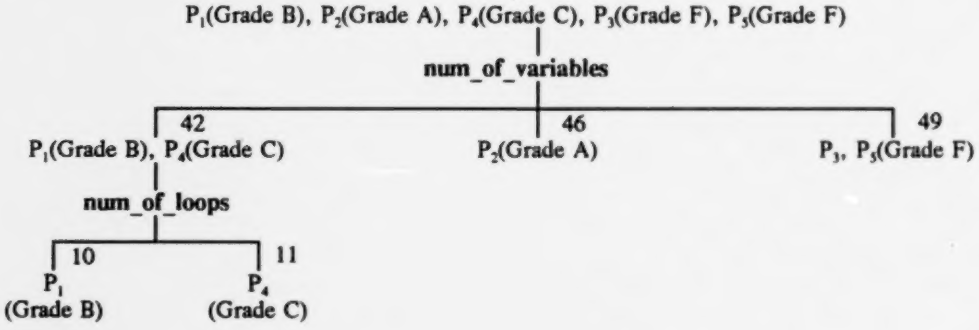


Figure 1. Resulting Decision Tree.

	Attributes													
Programs	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}
p_1	A	44	12	10	14	7	29	9	27	19	7	8	0	0
p_2	B	42	14	10	16	7	38	9	34	23	7	8	0	0
p_3	A	46	14	11	16	7	34	9	31	20	7	8	0	0
p_4	D	34	13	13	12	5	19	6	23	9	5	7	0	0
p_5	A	46	14	11	13	7	29	9	25	18	7	8	0	0
p_6	F	48	18	11	13	7	32	9	28	18	7	11	0	0
p_7	C	41	15	11	4	6	19	8	19	1	6	7	0	0
p_8	F	49	20	11	13	7	32	9	28	19	7	11	0	0
p_9	C	34	15	7	16	5	40	10	55	15	7	8	0	0
p_{10}	A	50	16	11	16	9	44	8	41	22	9	10	0	0
p_{11}	A	46	18	12	15	7	43	10	47	21	7	8	0	0
p_{12}	C	46	26	11	32	5	90	8	90	17	6	12	0	0
p_{13}	C	60	23	9	15	7	67	10	54	22	7	10	0	0
p_{14}	D	24	57	46	26	3	118	10	113	9	3	6	0	0
p_{15}	C	48	22	11	18	6	89	13	66	18	11	6	0	0

Table 3. Knowledge system generated for a student assignment.

class within a leaf are compared. The *modified* vectors for the two possible cases of plagiarism are:

p_6	F	48	18	11	13	7	32	9	27	18	7	11	0	0
p_8	F	48	18	11	13	7	32	9	27	18	7	11	0	0

---

p_9	C	34	15	7	16	5	40	10	55	15	7	8	0	0
p_{13}	C	60	23	9	15	7	67	10	55	22	7	10	0	0

Since the entire vectors of p_6 and p_8 are identical, plagiarism is suspected with only this pair of programs and the instructor is so notified. The vectors of programs p_9 and p_{13} do not match, thus we must conclude that they were independently produced.

Obviously, if each leaf in the classification tree has only a single program, then no plagiarism is suspected. If after using all the attributes, programs with different classes end up in the same leaf, then inconsistency in the grading is suspected.

Attributes	Interval
num_of_variables (a_2)	[0]
assignment_statements (a_3)	[0..2]
num_of_loops (a_4)	[0]
num_of_decisions (a_5)	[0]
num_of_subprog (a_6)	[0]
total_operators (a_7)	[0..1]
unique_operators (a_8)	[0]
total_operands (a_9)	[0..1]
unique_operands (a_{10})	[0..1]
routine_calls (a_{11})	[0]
num_of_arrays (a_{12})	[0]
num_of_records (a_{13})	[0]
num_of_pointers (a_{14})	[0]

Table 4. The range of tolerance for each of the attributes.

To this point, 327 Pascal programs have been analysed. The Plagiarism Detection System identified 13 pairs and 3 triplets of programs in which plagiarism was suspected. After the class instructors inspected the suspected programs, it was found that all of the triplets were identical and plagiarism did occur. Among the pairs of programs, two pairs were misdiagnosed due to inconsistency in the grading. The other eleven pairs of programs were involved in plagiarism. It should also be pointed out that one other pair of programs which was involved in plagiarism was not detected by the system. In this case, the use of self-cancelling operations fooled the system.

Future Developments

One area where further development is needed is in making our counting algorithm more sophisticated to detect non-cosmetic alterations. These non-cosmetic alterations generally fall into one of six well-defined impurity classes (Bulut, 1974). Currently, our system will detect alterations which fall into two of these impurity classes, namely ambiguous usage of an operand and unnecessary replacements. However, we yet have to detect alterations which fall into the following classes: self-cancelling operations, synonymous usage of operands, common sub-expressions, and unfactored sub-expressions. We hasten to add that alterations in the above classes constitute sophisticated plagiarism, and generally sophisticated plagiarism is not the problem. Students in introductory classes who copy usually use rather simplistic methods to disguise their plagiarizing.

It is also our intention to increase the sophistication of the plagiarism detection system by identifying programs with the same programming style using an analysis of program features. Redish and Smith (1986)

	Attributes													
Programs	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}
p_1	A	44	12	10	14	7	29	9	**27**	19	7	8	0	0
p_2	B	42	14	10	16	7	38	9	**34**	23	7	8	0	0
p_3	A	46	14	11	16	7	34	9	**31**	20	7	8	0	0
p_4	D	34	13	13	12	5	19	6	**23**	9	5	7	0	0
p_5	A	46	14	11	13	7	29	9	**25**	18	7	8	0	0
p_6	F	48	18	11	13	7	32	9	**27**	18	7	11	0	0
p_7	C	41	15	11	4	6	19	8	**19**	1	6	7	0	0
p_8	F	49	20	11	13	7	32	9	**27**	19	7	11	0	0
p_9	C	34	15	7	16	5	40	10	**55**	15	7	8	0	0
p_{10}	A	50	16	11	16	9	44	8	**41**	22	9	10	0	0
p_{11}	A	46	18	12	15	7	43	10	**47**	21	7	8	0	0
p_{12}	C	46	26	11	32	5	90	8	**90**	17	6	12	0	0
p_{13}	C	60	23	9	15	7	67	10	**55**	22	7	10	0	0
p_{14}	D	24	57	46	26	3	118	10	**113**	9	3	6	0	0
p_{15}	C	48	22	11	18	6	89	13	**66**	18	11	6	0	0

Table 5. Modified knowledge system.

has shown that programming style can be satisfactorily analyzed or quantified at the syntactic level. Obviously, with our knowledge system the stylistic attributes considered must be quantifiable. To identify programs with the same style, it is our intention to consider such attributes as economy, modularity, structure, documentation, and layout.

Conclusion

This project has developed and implemented a plagiarism detection system to be used in computer science classes in which the programming assignments are done in PASCAL. However, with very little effort, the system can be modified to detect plagiarism in programs written in any structured, procedural language. The approach taken in this project has been to use the inductive inference algorithm, ID3, to classify programs in a decision tree. If a leaf of this tree has but one program, obviously no plagiarism was involved with this program. If a leaf has more than one program, then the entire knowledge vectors of these programs are compared. If the vectors are not equivalent, plagiarism is not suspected. If the vectors are equivalent and the programs are assigned the same letter grade then plagiarism is suspected. If the vectors are equivalent and the programs are not assigned the same grade, then inconsistency in the grading procedure is suspected.

The plagiarism detection system has been tested on a number of PASCAL programming assignments in an introductory computer science class. The results to date have been encouraging. It has enabled us to detect cases

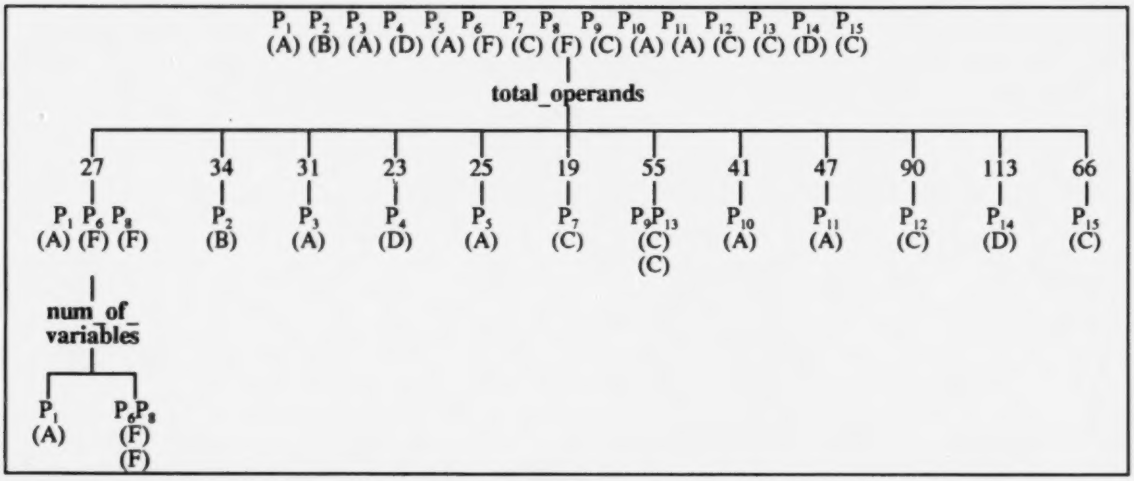


Figure 2. Classification Tree for the Student Assignment.

of plagiarism that had gone unnoticed by the graders. Together with this advantage, anecdotal evidence indicates that the students' knowledge of the existence of such a system is likely to have significant influence on discouraging program plagiarism.

References

Berghel, H. L. & Sallach, D. L. (1985) Computer Program Plagiarism Detection: The Limits of the Halstead Metric. *Journal of Educational Computing Research*, 1 (3), 295-315.

Bulut, N. & Halstead, M. (March, 1974) Impurities found in algorithm implementations. *ACM SIGPLAN Notices*, *9* (3), 9-12.

Donaldson, J., Lancaster, A., & Sposato, P. (1981) A plagiarism detection system. *ACM SIGCSE Bulletin*, *13* (1), 21-25.

Grier, S. (1981) A tool that detects plagiarism in Pascal programs. *ACM SIGCSE Bulletin*, *13* (1), 15-20.

Ottenstein, K. (1976) An algorithmic approach to the detection and prevention of plagiarism. *ACM SIGCSE Bulletin*, *8* (4), 30-41.

Quinlan, J. R. (1983) Learning efficient classification procedures and their application to Chess End Games. In *Machine Learning: An Artificial Intelligence Approach*, Michalski, R. S. et al., (Eds.). Palo Alto, CA: Tioga Publishing Co.

Redish, K. A., & Smyth, W. F. (Feb., 1986) Program style analysis: A national by-product of program compilation. *Communications of the ACM*, 29 (2), 126-133.

Robinson, S., & Soffa, M. (Feb., 1980) An instructional aid for student programs. *ACM SIGCSE Bulletin*, *12* (1), 118-127.

Shaw, M., Jones, A., Kneuven, P., McDermett, J., Miller, P., & Notken, D. (1980) Cheating policy in a computer science department. *ACM SIGCSE Bulletin*, *12* (2), 72-76.

Unethical "Computer" Behavior: Who is Responsible?

Lawrence G. Martin
Executive Secretary
Subcommittee on Automated Information Systems Security

We live in a society that believes that maturity and responsibility come with age. Most states require an individual to be a minimum age before granting certain rights and privileges. Most parents also set age limits for their children. The bottom line is that we as parents, and as society, feel a high degree of confidence that the individuals we've entrusted with certain privileges will behave in a mature and responsible manner when they are old enough to both understand and appreciate the value of a resource or the possible ramifications if a privilege is abused or misused.

The advent of computers has created a paradox for our society. We view the computer as an incredible learning tool and something our children must master in order to be successful in the future. In order to give our children a head start, we are introducing them to computers *as early as possible*. The paradox for our society is that while we would consider it unconscionable to put a second grader behind the wheel of a motor vehicle, we do, in fact, put second graders behind the keyboards of computers.

Based on our society's track record of putting age limits on privileges where irresponsible behavior can do public harm, one begins to wonder if computer use is a privilege that will someday require a minimum age mandated by a state or federal law?

As early as possible, we teach children basic rules that become the foundation upon which the rules of acceptable or ethical behavior for mature adults are built. This is the premise of a recent best-seller entitled, "All I Really Need to Know I Learned in Kindergarten" by Robert Fulghum.

As we have brought computers into the elementary school classrooms and begun teaching these young impressionable minds about them, I believe we have been negligent. We have ignored the teaching of those simple and basic rules that become the foundation upon which the rules of acceptable or ethical "computer behavior" for mature users are built. We are not instilling in these young minds that with computers, we are capable of doing bad things too.

While I believe we have a responsibility to teach ethical "computer behavior" to our young along with the computer training itself, it is a hard problem. It is *not* difficult to demonstrate how many of mankind's inventions that serve useful purposes can also be deadly and debilitating. Electricity, chemicals and explosives are but a few examples. Through visits to schools by firemen, the use of videotapes, and hands-on demonstrations in laboratories, young students can witness the dangerous effects of misusing and mishandling these items.

Similarly, I believe we must develop audio/visual tools and techniques that clearly demonstrate at an elementary level, the harmful effects of misusing a computer. I don't claim to know what these techniques should look like, but I challenge you, the educational computing community to utilize whatever research and development funding might be available for such a thing and to find the answer. We must instill in these young minds that computer users are both responsible and accountable for their "computer" behavior and the effects that it causes, the same as they are responsible and accountable for their everyday behavior. I believe we can start to do this by making an impression with things that they can relate to. For example, having an eight year old of my own, I have related the issue to him in the following ways:

1) His favorite Saturday morning TV show is Teenage Mutant Ninja Turtles. He checks the TV Guide and sees that it comes on at 8:00 am Saturday morning on channel 11. He gets up at 7:30 am, sets the channel to 11, and anxiously awaits the beginning of the program. At 8:00 am, a rerun of I Love Lucy comes on and disappointment is obvious in his face. "But Dad, the TV Guide said..." We have all experienced such an incident. We trusted the integrity of the TV Guide. I explained that is the same thing that happens to someone who accesses their own data on a computer after someone else who did not have permission to do so got into their files and changed their data. Its integrity is lost and the person is disappointed and hurt.

2) Similar scenario, again his favorite Saturday morning TV show is Teenage Mutant Ninja Turtles. He checks the TV Guide and sees that it comes on at 8:00 am Saturday morning on channel 11. He gets up at 7:30 am, sets the channel to 11, and anxiously awaits the beginning of the program. At 8:00 am, as the program begins, the TV screen goes black. The power has gone off because blocks away a drunk driver has crashed his car into a utility pole. My son is denied the ability to watch his favorite show because of someone else's reckless act. I explain that is the same thing that happens when someone introduces a virus or does some other reckless act that denies someone the use of their computer.

3) Different scenario this time. My son takes his baseball cards to school to trade with friends at the end of the day. He puts them in his locker. At the

end of the day, he goes to his locker and his cards are missing. How could this be? I ask him if anyone else knows the combination to his lock. He says he doesn't think so. I look at his notebook and see that he has written his combination on the cover so he wouldn't forget it. I explain that this is the same way that people break into computers when people write down their passwords so they won't forget them and they lose valuable data the same way he lost his baseball cards.

I know that this is not the only way to teach him, but providing examples that he can relate to at his age is effective. As a start, we can build similar scenarios for each age group. The point here is that when they can relate the victim, the consequences of such behavior are clear.

Just as we have begun a grass roots effort in the elementary schools to change other undesirable aspects of our society's behavior with "Say No To Drugs" and "No Smoking" campaigns, we must plant the seeds of acceptable "computer behavior" into the elementary school curriculum to begin building the foundation for professional ethics.

I believe an excellent place to start such a campaign is already in place with the grocery store chain "computers for the schools" promotions that have been successful around the country. In the Baltimore area, a local chain is working with Apple in a program called "Apples for the Students." Also, Safeway is working with IBM in a program called "Food for Thought."

I believe that these programs are the perfect vehicle to carry this message to the schools. For example, as schools redeem their grocery store receipts and begin accepting delivery of their new computers, printers and software, the grocery chain could invite the schools to voluntarily participate in a "Commitment to Responsible Computing Program." By doing so, the schools would agree to incorporate the teaching of responsible computer use into their curricula. In this way, the grocery chains can play a major role in heightening the awareness of both the general public and our educational systems of the need for teaching "ethics in computing." The benefit would be invaluable.

The real benefit of the concept is that if a computer vendor launched such a program as "Commitment to Responsible Computing," they would only reach their customers and the people in the profession. The grocery chains, through their customer bases, can reach a real cross-section that includes many parents who are *not* computer literate and who *need* to be aware of what their children can do with computers. The grocery chains, through the schools and their PTA's, can help to bring the parents and their children together in understanding responsible computer use. In that way, the parents' role in the program *does not end* after their sales receipts are turned in.

While what I am suggesting seems a logical extension of the grocery chain programs, it need not be dependent on the grocery chains. The schools do not have to wait to be invited to participate in such a program but could on their own initiative develop and implement such as program.

What I am suggesting does not require anything more than a certificate suitable for framing and hanging in the school. It could read as follows:

COMMITMENT TO RESPONSIBLE COMPUTING

As our society becomes more and more dependent on the use of computers, the misuse and abuse of computer systems poses a threat to our way of life. We recognize that we have an important role in educating our students in the proper use of computers. Therefore, be it known to all who read this, that

____

has made a commitment to teaching responsible computing to our students. As part of this commitment, we will strive to:

1) teach students that computer users are both responsible and accountable for their "computer" behavior and the effects that it causes, the same as they are responsible and accountable for their everyday behavior,
2) demonstrate to students the harmful effects of misusing a computer,
3) enlighten the parents of our students as to what their children are capable of doing with computers, and
4) bring the students and their parents together in understanding responsible computer use.

____

Principal

Such a certificate could be furnished by the grocery chain, the vendor or the school itself.

In an optional second phase of the program, the schools could then either themselves or through a sponsor, work to recognize the individual students who make the commitment to responsible computing with a button and/or ribbon and a certificate. Not only would the certificate and button and/or ribbon motivate the student to make the commitment and in the process heighten his or her awareness, but the certificate taken home by the proud student and shown to his or her parents would further heighten their awareness to the problem and their role in the solution. In addition, students proudly wearing the button and/or ribbon would serve as a model to encourage other students who either have not or are thinking about making the commitment.

Again, I share with you this thought as one way to get a positive reinforcement program started. I hope it at least stimulates some creative thinking as to some other ways.

I would now like to refocus your attention away from the schools and to the world beyond the ivy walls. Unfortunately, most of the computer users in our society and work force did not have benefit of formal computer training.

I speculate that many computer users had their first experiences with hi-tech electronics with computer or video games. Think about the first time you played Pac Man or Space Invaders. There were no rules posted showing how to play the game. You simply dropped your quarter in the slot and the game started. You learned the rules of the game by trial and error, pushing levers and pressing buttons.

The vendors of the pay-as-you-play games have a strong motivation for this strategy. The longer it takes a player to learn the rules of the game, the more quarters the player will deposit attempting to conquer the machine and meet its ultimate challenge. Challenge is a very strong motivator and it becomes addictive. It's a scary thought, but the strategy of *some* of these vendors may not be too unlike the strategy of the drug dealer who benefits from the weakness and addiction of others.

A segment on the ABC News Magazine show "20/20" featured one of the most recent video game crazes, Super Mario Brothers II by Nintendo. A more classic example of trial and error would be hard to find. The game is full of surprises and most are discovered by accident. The challenge is so great that often the lessons learned by a player the previous day dominate conversation on the school bus or at the lunch table in the school the next day. There is even a newsletter that describes the discoveries of others.

While their motive is profit, I believe that vendors of video and computer games have a responsibility to study their role in influencing the behavioral development of their users because I believe that for *some*, this "trial and error process" evolves into the mindset that whatever the games allow you to do is within the rules and is, therefore, acceptable. This mindset can then carry over to the use of computer systems.

Whether or not you agree with my conclusions and assumptions, you will probably agree that much of the responsibility for prevention and detection of this behavior falls solidly on the shoulders of the system manager, designer and/or security officer. It is a given that if these management officials are negligent or overlook something as simple as changing the preset system passwords, then a user, authorized or unauthorized is likely to find it and pursue it.

I observed this first hand when I was an employee of another Federal Agency over ten years ago. An employee in a local office found that the system allowed her to make corrections for claimants who were erroneously indicated as deceased. When a claimant would come into the office to find out why they had not received their last check, she would access their record. On occasion she would discover that due to a data entry error, the system had terminated their benefits because the claimant was deceased. She would then enter a special "resurrection transaction." The transaction would issue a one-time retroactive check back to their "date of death". After executing a number of authorized "resurrection" transactions, the employee devised a very clever scheme. She went to cemeteries and made a list of people who had been dead for 3 to 5 years, and then executed simultaneous resurrection and change of address transactions. The one-time retroactive checks were sent to her Post Office box. Once she received the checks, she'd "kill" the people off again and change the addresses back to what they had been. She was doing what the system allowed her to do. She was eventually caught through the check cashing process. However, today with direct deposit and electronic funds transfer, computer behavior such as this may go undetected indefinitely!

Why did the system allow her to do this? Put yourself in the place of the system designer for a minute. In figuring all of the possible scenarios, what is the likelihood that someone is going to come in and complain that they haven't received their last 60 benefit checks? The system developers were obviously very insightful to recognize and build into the system a way to resurrect and retro-actively pay those who were erroneously deceased. However, since most beneficiaries live from check to check, they never conceived of the possibility that someone would show up months or years later, and therefore did not build in the necessary controls that would prevent the transaction from executing when the time period or dollar amount was unreasonable. The employee simply abused the system by doing what the system allowed her to do.

We are all familiar with Murphy's Law. If it's possible for an operational system to have a flaw in it, then it will. These officials must recognize that they may, in many cases, be dealing with users, both authorized and unauthorized, who are of the mindset that what the system allows them to do is acceptable or who know what they are doing is wrong but behave unethically by taking advantage of a system's weakness.

In addition to finding and correcting system vulnerabilities, another deterrent may be the posting of electronic "No Trespassing" signs to caution all users at the beginning of their session. No Trespassing signs as we know them protect physical property generally citing a law and issuing a strong warning as to possible penalties and/or impending danger. While there is no impending physical danger associated with trespassing into computers, if the user does find a vulnerability in the system, he or she has already been put on notice that certain behavior is unacceptable and perhaps unlawful. If prosecuted, the user's intent can be more easily established because they can not claim that they did not know that what they were doing was wrong.

You may remember the case back in 1983 of Neal Patrick, the teenage member of the 414 hacker club in Milwaukee. Club members penetrated numerous computer systems including the Los Alamos National Laboratory and the Sloan-Kettering Cancer Institute.

Later, when asked by a Congressman, *under oath*, at what point he realized that what he was doing was wrong, Mr. Patrick responded, "when the FBI was knocking at my door."

In addition to testifying before Congress, something very few of us get to do in our lifetime, Mr. Patrick also got his picture on the cover of *Newsweek*, something even fewer of us achieve in our lifetime. And as the story goes, at that time, the movie "War Games" was popular and, as a publicity stunt, a local theater manager where "War Games" was showing, paid Mr. Patrick to sign autographs for movie-goers in front of the theater. All in all, quite an accomplishment for a 17-year old. As a model for other teens, this incident was more of an incentive rather than a deterrent.

Should our society make these people instant celebrities and glorify what they did or should we punish them and let that serve as an example to deter others?

Again, we have a paradox. While it is traditional of our society to bestow punishment befitting the crime, the criminals in most of these cases are young intelligent students who represent some of our most brilliant minds in our country. It is these brilliant minds that we will rely upon to carry this country into the next Century.

Therefore, do we lock up these young intelligent students in prison with murderers and bank robbers or do we find a way to channel their endeavors into more positive and productive activity?

We must recognize that a new criminal element is emerging that has some of the familiar characteristics, but is in a class by itself. In the same way that the *use* of computers has required change to the way we as a society function, so too, has the *misuse* of computers required change to the way we as a society deal with those who operate outside the established rules.

This thought raises a number of interesting questions. Are we making these changes timely and proactively. Are we waiting for computer misuse to occur and then reacting by passing laws that makes unethical computer behavior a crime with penalties that are not well thought out? Is the news media helping or hurting our efforts to deter computer crime?

I created another version of what the cover of the Neal Patrick issue of *Newsweek* could have looked like. Which do you believe would have been more effective as a deterrent to the would-be hacker?

You may recall in November 1988 when the INTERNET Worm brought the network down. Every television network and every major newspaper and magazine carried that as its feature story for days. Yet on February 15, 1989, the day after Herbert Zinn, Jr., the 18 year old hacker who penetrated AT&T and NATO systems and illegally copied and distributed more than $1.2 million dollars in software, was convicted and sentenced to nine months in federal prison, a $10,000 fine, and two and a half years probation, a two paragraph article appeared on the front page of the Business Section of the local newspaper. Mr. Zinn was the first conviction under a recent Federal Law, known as the Computer Fraud and Abuse Act of 1986. This was a true landmark in the computer security chronology and it went practically unnoticed. I realize that the penalties received are not as sensational and won't boost sales or ratings, but highlighting prison sentences, fines and other negative results experienced by the perpetrators *as well as* the damage, pain and suffering of the victims caused by computer abuse and misuse should at least make a potential hacker or computer criminal think twice before acting.

The victims of computer crimes often do not prosecute for fear of the adverse affects of the negative publicity. This failure to prosecute contributes to the temptation of would-be computer criminal because it lessens the fear of reprisal. For the individual who may be contemplating a computer crime, it may make the difference between right and wrong. Therefore, I add *victims* to the list of those responsible for deterring computer abuse and misuse.

However, the list is not complete without including parents. With the many everyday pressures from job and family, many parents might relax and find comfort in knowing that their child is nice and safe in their bedroom quietly "fooling around" with their computer, instead of getting in trouble by hanging out with the wrong crowd at the mall or on the street corner. What they may not realize however is that their child has a modem and may be attempting to penetrate other computers or networks. Obviously, Mr. Patrick's parents and Mr. Zinn's parents were unaware.

If a child had a gun in their possession, then their parents would surely want to know about it and take it away before someone was seriously injured or killed. Why should it be any different with a modem? If Mr. Patrick had changed patient treatment plans at the Sloan-Kettering Cancer Institute, someone could have been seriously injured or killed. Many teenagers have their own money and have the ability to make their own purchases. So it is not unreasonable that a child could have a modem and the parents not know. There may be many parents who are not computer literate and even if they knew their child had a modem would not know what a modem is and what it does. Should we require licenses for modems as we do for ham radios? Should we restrict the sale of modems to those 21 and over unless signed for by a responsible adult and then only after that adult has read the risks associated with having a modem in the home?

The introduction of computers and other hi-tech electronic equipment into the home brings with it even more responsibility for the parent to find out what it is and what it is capable of doing. The bottom line is that parents have a responsibility to know what their children are doing and to help the children understand what is acceptable behavior and what is unacceptable.

We've all heard the cliche "Do as I say and not as I do." We teach our children by example and they tend in turn to emulate our behavior. Our hi-tech world makes it very easy for us to serve as bad examples for our children without us even thinking about it. How many of us have copied a video tape on our VCR's even though it has an FBI warning at in the beginning. How many of us have made bootleg copies of copyrighted computer software? Why did we do it? Simple, because we didn't want to pay for it. Yet if we walk into a store, take something and walk out without paying for it, we are stealing. But if we copy a video tape or computer software, we do not think of it as stealing. Perhaps it is not as blatant, but someone ultimately suffers from the loss of the proper purchase of whatever we copied. We've all heard the rationale that surely the loss of one sale will not bankrupt a large corporation. Are our ethics contingent upon the net worth of the potential victim? Have we become modern day electronic Robin Hoods? We take from the perceived "rich" (owner of the copyright) and give to the perceived "poor" (ourselves or our children). Although we act with the best of intentions, the intangible repercussions of our actions may be ultimately hurting our children. When they observe this behavior, at the very least they may be confused by what appears to be a double standard. It's very similar to the confusion of children of parents who smoke. The schools are teaching that smoking is bad and unhealthy. When children see a parent smoking, they must think if Mom or Dad isn't worried about it, then why should I? So how can parents convince their children not to behave in a certain way when they themselves are behaving that way? "Do as I say, not as I do" just doesn't work in 1989!

Who can say how far they will take the examples we set for them today, as technology improves? Things we can not even dream of today will be achievable 20 years from now when our teenagers are in the workforce? We must be conscious of our own actions and realize that we are setting a precedent for the future. So yet another piece of the unethical computer behavior puzzle is the parents, who must recognize their role in the ethical development of their children. I believe that we as a society need an awareness raising as to the threats to our ethics by technology. We need to examine these subtle capabilities made possible by technology that allow us to deviate unconsciously from our normal ethical behavior. We also need to raise the public awareness of computer cause and effect.

While the schools, the systems managers, designers and security officers, electronic game vendors, the news media, the victims, the parents and society as a whole share in the responsibility for the computer user's behavior, the ultimate responsibility to behave in an acceptable manner belongs to the user.

We can draw many parallels between acceptable social behavior and acceptable "computer" behavior:

Unacceptable Social Behavior	Unacceptable "Computer" Behavior
1. To knowingly infect another person with a communicable disease.	To knowingly infect another person's computer with a virus.
2. To enter another person's home or drive another person's car without their permission.	To enter another person's system without permission.
3. To rummage through another person's belongings.	To rummage through another person's database(s).
4. To shoplift or take things belonging to another.	To make copies of copyrighted software.
5. To keep extra money if a store clerk gives us back too much change or to pay a lower price because merchandise is priced wrong.	To do things we know are wrong because the system allows us to.
6. To cause another person bodidy harm or death.	To cause someone denial of the use of his or her computer.

It takes a long time to change society's behavior. Drunk driving is a perfect example. With all of the attention it has received in recent years, there were still over 34,000 drunk driving arrests in Maryland in 1988. That's nearly 100 per day or one every 15 minutes. While the number of drunk driving arrests is on the decline, one every 15 minutes is a very real indicator that the problem has not gone away. And keep in mind that the 34,000 were only the ones that got caught.

I believe that *drunk driving* and *computer misuse* are the crimes that will reflect the 80's. It was this decade that raised the public awareness to the negative repercussions of both and the parallels are quite interesting. They both represent behavior which was tolerated until the potential dangers were realized. As awareness grew, tolerance lessened. Behavior that was once considered prankishness or mischievous is starting to be considered malicious and a criminal offense. How to deal with these new type criminals who are not typical of the stereotype criminal element has become an issue.

In Maryland there was a recent proposal to build a separate correctional facility just to house convicted drunk drivers who are given prison sentences. Our prisons are already overcrowded and adding 34,000 drunk drivers would only add to the problem. Such a facility would not only prevent the strain on the prison system but would segregate the drunk driver from the hardened criminal. Should we do the same for the convicted computer criminals?

While many consider the computer security problem and solution to be technical, I believe that the computer security problem is a people problem with both a technical and a people solution. As we all know, it is not a perfect world and the teaching of "computer ethics" will not eliminate unethical computer behavior. There will always be some, who although they know the rules, will disobey them.

It is only when we can't trust the behavior of the computer user, that we must have mechanisms and assurances that allow us to trust the system instead. The title of my paper asks, who is responsible for unethical computer behavior? I hope I have started each of you thinking about how all of us, in our various roles, share in the responsibility for and the consequences of unethical computer behavior.

Intelligent Hypertutoring for High School Science Instruction

John R. Bourne, Chuck Kinzer, Herman van der Molen,
Jerry Haden, Dona Mularkey, and Steve Schreiner
Vanderbilt University

Carlo Hyde, Roy Culbertson, and Peggy Guy
Martin Luther King Jr. Magnet High School
Nashville, Tennessee

Abstract

This paper describes an intelligent hypertutoring system and the use of the system in teaching high school science curriculum. Intelligent hypertutoring is a combination of intelligent tutoring systems and hypertext methodologies. A computer based system has been constructed that provides instructors with the ability to author complex instructional systems. Based on Smalltalk-80, the system provides both authoring and instructional delivery tools and runs on a variety of hardware platforms including the Macintosh and PC-386. The capabilities of the system and experiences that have been accumulated using the system during the 1989-90 school year will be discussed and a prospectus given for further development.

Algorithms for Korean and Chinese Language Keyboard Input in Microcomputers

Dennis Bilgin
Defense Language Institute

Abstract

Effective foreign language instruction by means of microcomputers requires the integration of foreign language orthographies with software tools that are routinely used for the creation of foreign language courseware. Courseware implementations in languages such as Korean and Chinese entail the creation of algorithms that modify the configuration of the standard ASCII keyboard. The design and creation of such algorithms involve the close collaboration of linguists and educational technologists.

This presentation will demonstrate Korean language keyboard input and focus on algorithm design requirements for Chinese language keyboard input.

Computer Generated Discrepant Events—Teachable Moments in an Elementary Mathematics Classroom

Barbara R. Biglan and Anne Stanko
Shady Side Academy Junior School

Abstract

Problem solving has been cited as a goal of elementary school mathematics programs. In order to promote true problem solving activities, not rote repetition of heuristics, the computer is used to present discrepant events and provide a workplace for hypothesis testing.

The turtle graphics of Logo are used to examine regular polygon construction. The students work with a "Rule of 360 degrees" for construction on the computer. Off the computer, when the same angle measurements are used, the shapes are different. This event leads to much experimentation, editing of procedures, redrawing, and even lively debate. The group proceeds to share information, construct tables, find patterns, and discover the relationship of interior and exterior angles.

Order of arithmetic operations are discovered by having students solve several arithmetic problems using pencil and paper. Their solutions are checked by having the computer perform the arithmetic operations. When solutions of students and computer differ, the search for the order of operation proceeds. Often several different answers to one problem promotes students explaining their reasoning to others. The next step in this investigation allows the student to teach the computer how to solve problems to match his solution.

Several commercially available software programs are also used to stimulate mathematical reasoning without depending heavily on only computation skills. These programs included *Hotel Victoria* and *The Goodell Diamond Caper* from Tom Snyder Productions and *Hands-On Math* from Ventura Software Company.

Project CHILD: Integrating Computers into the Elementary School Curriculum

Sally Butzin
Florida State University

Abstract

Project CHILD is a research and development project designed to create a model educational program to move Florida's elementary schools into the 21st century. It is based upon an active approach to learning that integrates computer technology and hands-on activities into the curriculum areas of reading, language arts, and mathematics. The expected outcome is to have, by 1990, a fully developed model program and camera-ready materials to disseminate to elementary schools seeking change.

The project has been designed in three phases:

(I) Materials development (completed June 30, 1988)
Teacher's Manual
Student Passports
Learning Activities Guides, Grades K-5 for Reading, Language Arts, Mathematics

(II) Implementation, fieldtesting, and formative evaluation in two pilot schools with formative evaluation and on-going revisions (completed June 30, 1989)

(III) Continue implementation and fieldtesting with summative evaluation and camera-ready materials (to be completed June 30, 1990)

The Project CHILD model and supporting print materials address the following: restructuring the school, restructuring the curriculum, empowering teachers, empowering students, establishing healthy working environments, increasing motivation and time-on-task, building bridges beyond the classroom, and providing a cost-effective program to meet the future needs of schools.

For more information or to visit a pilot school, contact:

Mr. Jorge Ortega
Program Specialist, Instructional Technology
Department of Education
522 Florida Education Center
Tallahassee, FL 32399
(904) 488-0095 or Suncom 278-0095
or
Dr. Sally Butzin
Project Manager
CIDS, Florida State University
2003 Apalachee Parkway
Tallahassee, FL 32301
(904) 487-2054 or Suncom 277-2054

Hypercard Instructional Stacks: Jump-linear, Tree, Network, and Single-frame Examples Appropriate for the K-12 Classroom

Michael Land
Midwestern State University

Abstract

This group of stacks was developed as part of a larger project to explore ways to integrate the use of technology into the teacher education and K-12 curricula. One of the objectives of the stacks illustrated in this presentation was to develop examples of stacks that illustrate and serve as an instructional model for different types of stack organization for preservice and inservice teachers.

Hypercard Instructional Stacks: Science and Mathematics Ideas for the K-12 Curriculum

Steven Tipps and Michael Land
Midwestern State University

Abstract

This group of stacks was developed as part of a larger project to explore ways to integrate the use of technology into the teacher education and K-12 curricula. One of the objectives of the stacks illustrated in this presentation was to develop examples of stacks illustrating the development and use of hypermedia instruction in K-12 science and mathematics.

The Impact of Interactive Video Presentation on High School Student Stress Management Unit

Sharon E. Smaldino and Susan J. Koch
University of Northern Iowa

Abstract

Expanding the use of technology in classrooms as a means of enhancing educational experiences for students has been one proposed response to increased state educational requirements for health education in Iowa schools. During a stress management unit (a component of a secondary school health education course), progressive muscle relaxation was demonstrated through the application of the interactive video technique. A *HyperCard*/videodisc tutorial guided the students through the steps involved in the technique. As student interacted with the tutorial, a record of the amount of time spent on each screen and the number of times the video was viewed was automatically maintained. Responses to an exit questionnaire indicated students found the technology easy to use and expressed interest in learning more about both stress management and the use of interactive video.

Development and Delivery of Interactive Video and Computer Assisted Instruction at Columbia State Community College

Stephen L. Stropes and Deanna Naddy
Columbia State Community College

Abstract

The Center of Emphasis, instituted at Columbia State Community College in 1986, has a central focus of developing interactive video and computer assisted instruction (CAI) programs for students. Many of the programs housed within the Center were developed in collaboration with faculty members and produced within the Center's media production facilities. The uniqueness of the Center is the development of interdisciplinary interactive video and CAI programs that are tailored to the specific educational requirements of Columbia State faculty.

These programs are available to students in open labs located on and off campus as a supplement to traditional instruction. The staff of the Center encourages and assists faculty in the development of interactive media instructional programs. Although technical knowledge or previous experience are assets, they are not essential for faculty to develop these programs.

Columbia State has recently demonstrated its creative interactive programs at state and national conferences and expositions. The Center of Emphasis is available to assist other institutions involved with interactive video and CAI instruction.

This presentation will involve a brief description of the Center of Emphasis facilities followed by a discussion of the production and delivery of interactive video and CAI programs for student instruction at Columbia State Community College.

Building Bridges: A Report on Technology Development Sites for Speech-Language-Hearing Applications

Paula S. Cochran
Northeast Missouri State University

Julie H. Matherson
University of Mississippi

Frank G. Pagan
California State University, Fullerton

Glen L. Bull
University of Virginia

Abstract

This project is the result of an initiative entitled "Building Bridges" by the Partnership Grants program of Apple Computer's Community Affairs Department. As a result of a grant competition sponsored by the American Speech-Language-Hearing Foundation (ASHF), eight national sites were chosen to receive equipment and technical support from Apple Computer, Inc. for the purpose of developing innovative clinical software for speech, language, and hearing disorders.

The projects at each ASHF Technology Development Site have been underway for more than a year. They include the development of a wide variety of instructional and clinical applications that are summarized in this presentation.

The 1988-90 Technology Development Sites include:

- Baylor College of Medicine/The Methodist Hospital, Houston, TX
- Ithaca College, Ithaca, NY
- Northeast Missouri State University, MO & University of Virginia, VA
- Northwestern University, Evanston, IL
- University of Arizona, Tucson, AZ
- University of Mississippi, University, MS
- University of Nebraska, Lincoln, NE

For more information about the results of these projects contact Nancy Minghetti, American Speech-Language-Hearing Foundation, 10801 Rockville Pike, Rockville, MD 20852.

Special Touches: Adapting Computer Applications for Students with Special Needs

Christine L. Appert
University of Virginia

Abstract

Microcomputer technology offers a valuable medium for the versatile and individual planning required by students with special learning needs. The computer extends to the child a realm of control and provides the teacher with unlimited possibilities for customization and specialized instruction. Well designed software applications can compliment academic and developmental goals while offering a unique dimension to learning and independence.

"Special touches" lie in strategies that couple instructional objectives and learning needs with microcomputer use. Software selection is complicated by dilemma of accommodating student abilities and objectives with a match from the range of programs available. In reality, it is difficult to identify programs that incorporate all desirable qualities and exactly fulfill instructional expectations. Educators must often resort to ingenious tactics that sidestep program limitations and take advantage of the software's best features.

This presentation will provide an overview of practical techniques that have been found effective in creating flexible learning environments. Participants will receive suggestions for utilizing applications that adapt readily to unique individual needs and recommended resources.

Computer Assisted Instruction—It Takes a Plan

Linda Willson and Beverly Moore
Nations Ford Elementary School
Charlotte, North Carolina

Abstract

The presenters will discuss the planning of a program to use computer assisted instruction in an elementary school. The program includes changing a computer lab from a game-playing place to an instructional setting, correlating software to the curriculum, planning an inservice program for teachers, using one computer in a classroom, and networking a lab and classroom computer. Part of the presentation will deal with initiating change in a school setting, and part will deal with the progress students have made on standardized achievement tests over a 3-year period.

The presentation is made by two administrators with limited technical backgrounds who had a vision for how computers could be used to motivate, tutor, remediate, and expand instructional techniques in order to improve an overall school program.

Using Technology to Impact a District's Curriculum

Brooke P. Woods
Jackson Public School District
Jackson, Mississippi

Abstract

Realizing the need to address the varying learning styles of today's diverse student population, educators in the Jackson Public School District are ever on the alert for the newest and most creative ways available to meet the learning needs of their students.

That children in 1989 are stimulated by today's technology comes as no surprise to anyone, nor does that fact that "stimulated" children quite readily translate into "learning" children. However, for technology, as well as any other instructional tool, to impact the total program, it must be able to create its niche within the existing curriculum. And for use of a computer to effectively serve as instructional support to a district's curriculum, the software that is used must show a high correlation with district learning objectives. And finally, use of the technology must lead to an affirmative answer to the pivotal question, "Can it help us to accomplish our mission of teaching our identified curriculum to all our children in a better and more efficient way than we could without it?"

The spring of 1988 presented an answer to this question for Jackson educators in the placement of a computer based instructional system in the fourth-grade language arts and mathematics classes in one of the district's elementary schools. Five networked computers were placed in the language arts classroom; five networked computers were placed in the mathematics classroom; and a file server was loaded with language arts and mathematics software that had been correlated to fourth-grade learning objectives. The system's goals were to provide a classroom management structure to help teachers use current computer technology to teach basic skills, while at the same time supplying a framework for meaningful integration of curriculum content and computer software.

The teachers who were to implement the project had voiced initial misgivings and concern as to how the technology could fit into their total program and "do its thing" while still leaving instructional time intact for their own "tried and true" teaching routines. However, they soon realized that "technology" and "curriculum" are not mutually exclusive terms; that not only is it possible to integrate technology into an existing program, but that technology can actually enhance the curriculum as it provides for alternative teaching strategies to meet the wide range of learning needs of today's students.

Data collected from implementation of this project in the fourth-grade language arts and mathematics classes brought about unexpected, but gratifying, results. This has subsequently led to an expansion of the program into the fifth and sixth grades in this elementary school, as well as an initiation of the project in three additional schools in the district.

67

Learning Nursing Diagnosis via Real-Time Computer-Mediated Communication

Robert N. Higgins
Ontario Institute for Studies in Education
252 Bloor Street West, Rm. 3-346
Toronto, Ontario, CANADA M5S 1V6
(416) 926-4713 Bitnet: R_HIGGINS@UTOROISE

Abstract

Computer-mediated communication (CMC) is becoming accessible to greater numbers of people; "online" education is growing as a viable supplementary or alternative approach for teaching and learning. The research reported in this paper explores and describes the comparative effects of synchronous text-based CMC with asynchronous text-based CMC in terms of cognitive activity, cooperative processes, and the quality of outcomes. First and second year nursing students worked in pairs to establish a nursing diagnosis and related nursing care plan based on assessment information provided in a patient case study. Computers in a local area network were used to link the members of each pair. The only means of communication was via the keyboard and monitor.

The most general finding relates to the increased ease of mutual facilitation, motivation, and prompting demonstrated by the synchronous group; cooperative interaction more readily transpired. Cognitive activities such as arguing and debating were more evident in the synchronous pairs as well. Also, the synchronous pairs demonstrated better adherence to the principles and criteria that guide the development and statement of a nursing diagnosis.

Introduction

Computer-mediated communication (CMC), cooperative learning, and nursing education, are the main themes of this paper. Human communication is integral to all three. The computer mediates human communication by facilitating the entry, transmission, and storage of messages. This communication is asynchronous when the participants write and post messages which are left for viewing by others at another time. Synchronous CMC, on the other hand, refers to interaction that takes place in real-time, while the participants are simultaneously present at their computer workstations.

Now that computers are being used to mediate human communication for the purposes of education, a number of educators have reported their experiences. Some address issues relating to the use of computer conferencing (Davie, 1987; Harasim, 1987; McCreary and Van Duren, 1987; Phillips et al., 1988) in asynchronous mode. Others report on the application of synchronous communication via local area networks in the classroom (Sleightholm Cairns and O'Kelly, 1988; Wilton, 1987). For the most part, these are descriptive studies that focus on delivery, group process, student participation, and access.

The reports outline advantages and disadvantages relating to both the use of CMC, and the quality and features of the software that supports it. However, when recommendations are made, they generally do not take into consideration alternative modes of CMC or the integration of other media. Suggestions are made primarily in terms of the medium and the mode that is familiar.

Those involved with computer conferencing seem particularly unenthusiastic about possibilities for a synchronous adjunct. This is not surprising in view of the fact that the features and capabilities provided by computer conferencing software have not changed significantly over the past 10 years, and reasonable synchronous capabilities have not been readily available. Another reason is that the asynchronous nature of computer conferencing is regarded as one of its most valued attributes. Levinson (1988) notes, " 'Asynchronous' or nonimmediate communication ...may produce exchanges of richer intellectual quality than those resulting from immediate face-to-face dialogue" (p. 115).

With these issues in mind, it seemed appropriate to undertake a study to compare the effects of synchronous versus asynchronous modes of computer-mediated communication. Currently, the research, reports, and reviews of CMC in education do not adequately address learning in terms of cognitive processes. The social interaction and relational approaches demonstrate support for learning through greater and more equitable participation, access to the knowledge of others, and independence of time and place (Harasim, 1989); however, little effort has been made to investigate the cognitive activity of cooperative learners using this medium.

Reports in the cooperative-learning literature indicate that cognitive activity can be enhanced through student-student discussion (Johnson, 1981; Powell, 1986). "The processes that promote higher achievement...among students may include the promotion of high-quality reasoning strategies, the constructive management of conflict over ideas and conclusions, increased time on task, [and] more elaborative information processing" (Johnson and Johnson, 1985, p. 120). Also, for users of CMC, synchronous interaction may be more powerful for problem-oriented discussions (Beckwith, 1987). Generally, real-time feedback and prompting in synchronous mode may bring about a higher order of interaction, improved products, and greater satisfaction.

In this study, there were a number of reasons for using a nursing student population. First, nursing has a need for innovative educational practice to meet upgrading and continuing education requirements for members of the profession who may be located in remote or semi-remote areas of the country. Second, nurses, like other professionals, are having to integrate computers with many aspects of their practice. Third, communication skills are a professional requirement in nursing and the students receive specific training in the areas of group process and interpersonal communication. Fourth, the students in this sample, as a normal component of their curriculum, had experience with a computer-assisted instruction program used to introduce them to nursing diagnosis (Yoder, 1984). This was expected to help reduce variation in the subjects' acquaintance with computers prior to participating in the study.

The focus on nursing diagnosis was important because it has become an essential element of nursing education, the nursing process, and indeed, nursing professionalism. Nurses use nursing diagnoses to state clearly the actual or potential health problems affecting their client's physical, psycho-social, and spiritual well being. This holistic approach to each client, regardless of their medical diagnosis, helps distinguish nursing roles from those of medicine and the other health professions. The nursing diagnoses direct attention to those aspects of a client's condition that can be helped by nursing interventions, and "which nurses, by virtue of their education and experience, are capable and licensed to treat" (Gordon, 1982, p. 2).

Computer-Mediated Communication

Computer conferencing arose as the first generalized means of mediating human communication via computer. The computer is used to accumulate and store notes or messages in files that are accessible to designated groups of people. Members of any such group can use this many-to-many messaging system to facilitate group interaction or conferencing. They can share ideas by adding their own messages and by reading those left by other members of the group.

As the number and kind of conferencing systems increased, the need for assessment and status review became evident. Kerr and Hiltz (1982) surveyed the most prominent CMC projects of the time by soliciting opinions and observations from key representatives of the groups and organizations utilizing this technology. In line with much of the research preceding their study, and much that would come after, a fairly standard set of factors and effects were analyzed. These included system software design factors, and factors affecting user acceptance of new systems. (p. 174-175)

Kerr and Hiltz also gathered information about the cognitive impacts of CMC on individuals and groups. They reviewed the existing literature, and used reports from evaluators, to find support for their expectations:

> The overall pattern suggests that the more socially significant cognitive impacts, such as those including conceptual skills and learning, generated support, whereas those which may be more trivial, such as spelling and typing skills, and those which are clearly negative in impact, such as information overload, are much lower on the list [*i.e.*: they were less verifiable from the sources] (1982, p. 102).

Educational Computer-Mediated Communication

The potential applications and benefits of CMC in education must have revealed themselves from the beginning. Hiltz and Turoff (1978) associate this potential with "the trend to life-long learning" (p. 191). For example, the participation of professionals in conferences oriented to the technical or scientific issues of their field, is in itself a form of continuing education (p. 195).

Later works by Hiltz are more specifically directed to educational issues. Appropriate and effective structures and methods are sought for educational practice in the "virtual classroom" (1986, p. 97). In particular, Hiltz leans toward cooperative learning when she states that, "interactive computer use can lead to greater communication among the members of a learning group or 'class,' not just between teacher and student" (p. 96). Increased interaction, and the facilitation of "active learning" become the primary variables by which effectiveness of the virtual classroom can be determined.

Although "active learning" is not clearly defined in terms of specific learning or cognitive outcomes, it is reasonable to assume that, since the activity is human communication in an educational context, the exchange of ideas, the presentation and defense of arguments, and the integration and response to feedback are acceptable evidence of a learning process. A less tenable extension, however, is to suggest that learning can be assessed through unclassified measures of participation rates and the volume of individual contributions.

In a very short period of time, the number of online educational opportunities has increased dramatically. Most offerings are in higher education, with graduate studies predominating. For the most part, CMC has been regarded as a new medium of human communication. Many studies are oriented to this "New Media" approach (Rice, 1984). More recently, however, online education has been proclaimed as a new domain which:

> ...enables us as educators and as learners to engage in learning interactions more easily, more often and perhaps more effectively, but also to develop qualitatively new and different forms of educational interactions. (Harasim, 1989, p. 62)

Harasim (1989) notes that the key attributes characterizing this new domain are: asynchronicity, independence of place, and a many-to-many interaction (p. 50). Unfortunately, this scheme imposes severe limitations on the real potential of what remains an immature medium. It is true that the characteristics she describes are among the most frequently reported positive attributes, but generally, such reports reflect user enthusiasm for novel approaches, without adequate analysis of other possibilities. For example, many-to-many interaction is only one of the ways interaction can be structured in CMC. There are situations where one-to-one is necessary within the online context; special guidance, direction, or support may be needed by some individuals through private messaging. Independence of place is not entirely critical either. Educational activities using computer conferencing or its successors may well prove as beneficial for those who all attend the same institution, as for those more geographically dispersed. Finally, time independence (asynchronicity) takes into account only one of the possible modes of communication in CMC. The insufficiency of asynchronous interaction has been described in regard to group problem-solving (Beckwith, 1987). Harasim, as well, alludes to the possible need for a synchronous group communication facility (1989, p. 61).

Research Design and Methods

The quality and nature of learning interactions in this new environment are affected by the techniques used to facilitate communication among participants. Two techniques were compared in this study. Eighteen pairs of students were divided into two groups; nine pairs used the synchronous mode and nine pairs used the asynchronous mode. All pairs had two hours of online time to complete the task of establishing a nursing diagnosis and related nursing care plan based on assessment information provided in a patient case study.

A local area network was used to mediate the interaction. The software allowed the asynchronous pairs to write notes to each other and save them in a common, sequential message-base similar to most computer conferencing systems. The software also provided a "phone" utility in the same windowing environment that allowed the synchronous pairs to write to each other, character by character, in real-time. Members of the asynchronous pairs attended for one half-hour on 4 separate occasions; each time they would read and respond to the notes left by their partner. The synchronous pairs attended once for a two hour session of mutual interaction. Members of the pairs remained anonymous to each other throughout the study.

The nursing diagnosis task was particularly well suited to resolution through cooperative discussion. As part of the curriculum in nursing theory, students learn the principles and criteria used to establish a nursing diagnosis. However, in their first and second year they are not required to use exact, or officially accepted wording. Instead, they are encouraged to use their own words while following a set of guidelines. This flexibility opened the door to deliberations concerning both the form and the content of nursing diagnosis. The content was based on significant data provided in the case study and identification of factors which are within the realm of nursing practice. Form or format refers to the specification of the components of the nursing diagnosis statement, and the proper sequencing of those components. The basic PES model is as follows:

"(P)roblem" related to "(E)tiology"
as manifested by "(S)igns and symptoms"
(Iyer, et al., 1986, p. 83)

For example, one of the nursing diagnoses for a patient with a medical diagnosis of "stroke" might be:

"impaired mobility" related to "left-sided weakness"
as manifested by "patient limps and reports
a feeling of numbness in the left hand"

Data Analysis

All messages, whether synchronous or asynchronous were captured in log files during communication activities. The analytic approach taken was to examine the content of the messages as opposed to examining the sources and interrelations of the messages. In communication research, this characterizes a content analysis as opposed to an interaction, relational, or even a hypertextual analysis.

Numerous categorization systems for the analysis of interpersonal or group communication exist, but most are oriented to interactions, relations, and other social factors (Bales, 1950; Danziger, 1976; Hirokawa, 1980; Rogers and Farace, 1975). Although the social-relations aspects of group problem-solving are considered in some of these reports, few deal specifically with the cognitive activity of the participants. Powell (1986), on the other hand, developed categories for, "...the cognitive content of the verbal interaction" (p. 26) in his studies of tutored versus leaderless discussion groups. His categories and criteria were adapted for the coding of cognitive content in the current study. They include: "giving an opinion", "giving information", "arguing", "asking for information", "clarifying", and "formulating problems". Of these, "arguing" is deemed to be the more cognitively intense because it requires integration and understanding of the opinions, issues, and problems under discussion. "Arguing" is distinguished from "clarifying" which is used to elaborate and explain an opinion, but which does not need to involve integration of an opposing position.

n5:
Well since the root of his problems focus on the depression and anxiety, and his ineffective coping with these feelings, I would say deal with his self-image first because you have to resolve this before you could tackle the nutrition and sleeping problems. What do you think?

n6:
I agree. But did you think that ineffective coping related to anxiety and depression was inappropriate? I believe it is a legitimate diagnosis meaning ineffective coping with everything in general.

n5:
Good point. Could we put ineffective coping with activities of daily living related to feelings of depression and anxiety?

n6:
Sounds terrific! Wait a minute while I write that down! I also think this should be the first one we work on since as you pointed out it is the root of all his problems.

n5:
Super. What diagnosis would be next in line?

Figure 1. Synchronous Discussion with Arguing and Facilitation Aspects

Powell put less emphasis on group processes than most other analysts so his scheme contains only one category for all such activity whether negative or positive. This is much too narrow in light of the enormous work done in social psychology concerning the effects of interaction. Beckwith (1987) recognizes the importance of "debilitative" and "facilitative" interaction as determinants of success in group problem-solving (pp. 101-104). Therefore, to take account of such interaction, these components of Beckwith's model were adapted for the coding of cooperative content.

In addition to the content analysis, the steps and process used to establish nursing diagnoses were assessed in relation to the principles and guidelines that were familiar to the students. Some of the criteria included: appropriate use of data from the case study, proper ordering of the components of the nursing diagnosis statement, and acceptable use of connecting terms (eg: related to, as manifested by).

Findings

Although the total amount of content categorized as cognitive activity was the same for both groups, twice the amount of interactive arguing appeared in the transcripts of the synchronous pairs than those of the asynchronous pairs. Without the immediacy of synchronous interaction, the asynchronous pairs were less likely to frame opinions and explanations in a manner that invited feedback or correction.

In terms of cooperative activity, the synchronous pairs demonstrated a larger portion of facilitative exchanges. These included efforts by the participants to make each other more comfortable as well as clear statements of support, understanding, and encouragement.

n12:
Okay. Is depression classified as a nursing diagnosis?

n11:
I don't think so because it is a medical diag. I can't remember how we are supposed to state it. Maybe self image deficit.

n12:
How about hopelessness based on the things he said have changed in his life and the way he sees himself.

n11:
Great! Shall we put it into their format? Hopelessness related to, as manifested by, etc.

n12:
Thats good, but I'm not sure exactly what to say its related to. There are a lot of things it could be manifested by.

n11:
How about related to perceived failure? Or, I can't think of a better word for perceived incompetence.

n12:
I like the perceived failure. It indicates how he feels and can result from his parents high expectations.

n11:
Should we specify what he feels he failed at? Can't really say failed at life.

n12:
No we can't put that. In paragraph 3 it says he can never seem to satisfy his parents. Maybe that has something to do with it.

n11:
So, perceived failure in meeting expectations of parents. I don't know if that is okay, if it would be an approved etiology.

Figure 2. Synchronous Discussion with Attention to Principles and Criteria

n22:
Patient seems to be experiencing severe anxiety related to his low self-worth. Although his eating and sleeping habits have deteriorated, I believe his depression should be addressed first. If we can get him to talk more about the circumstances surrounding his attacks, I believe we will have a better chance at ameliorating his deteriorating eating and sleeping habits. Patient should be teamed up with a psychiatrist and possibly a therapy group.

n26:
I agree that his depression is a major priority and should be looked after first. The only way to overcome the depression would be for him to talk to someone about it. The therapy would in all likelihood do him good. I suggest that group therapy (after the initial one-on-one with the therapist) where people with problems like his and some that are worst (to give the perspective that there are bigger problems than his) so that he has a wall he can lean on for support.

n22:
Patient doesn't seem to have problems communicating openly to staff, perhaps it would be good to observe his communication with his parents and his best friends. I think it would be wise to talk to his parents and friends separately and get their input as to when the depression started and how bad it has been. An idea might be to get the patient to open up about what he likes and what he is capable of doing. While the patient is in the hospital, it might be good to get him to participate in some activities.

n26:
Agreed. Find out about his families standards of excellence verses his own abilities in school and out. Try and determine why there is such a difference.

Figure 3. Asynchronous Discussion

Figure 1 gives an excerpt from one of the synchronous discussions which demonstrates the arguing and facilitation aspects.

One of the more interesting findings concerns attention to, or adherence to, the principles and criteria that guide the development and statement of a nursing diagnosis. The synchronous pairs spent more time debating the correctness of their statements relative to the model and guidelines they had learned. Asynchronous pairs were less concerned with such formal structure. They concentrated more on analysis of the data provided in the case study, and they questioned the form or components of their stated nursing diagnoses less frequently.

Figure 2 gives an excerpt from another synchronous discussion which demonstrates attention to principles and criteria.

The next excerpt, given in Figure 3, is the first four exchanges from one of the asynchronous discussions. It demonstrates a decreased amount of arguing, facilitation, and attention to principles and criteria. Instead, the pair begins to discuss nursing interventions without adequately formulating and stating nursing diagnoses.

Discussion

The excerpts shown in the previous section are some of the most obvious examples of the differing content and approaches identified in this study. Since the author was the only coder, interpretive bias cannot be discounted. Most of the transcripts could not yield such clear cut examples as those provided above. Further, it would not be valid to claim that these findings explain the phenomenon.

The more frequent use of facilitative comments by the synchronous pairs, for example, might be easily explained by the fact that the number of individual exchanges was considerably greater in synchronous mode. On average, the synchronous pairs were alternating input after every two sentences while the asynchronous participants had to fill up approximately three paragraphs before their partner would read and respond. As such, an increase in motivating prompts, encouragement, and other facilitative expressions would soon appear excessive in one asynchronous contribution.

Similarly, arguing by the synchronous pairs may be influenced by the fact that each participant knows that an immediate response is possible and probable. There is no reason to let a doubtful stance on an issue pass if clarification or support for the argument can be promptly elicited. The asynchronous participants, on the other hand, realize that they may be able to pursue certain questions on their own, between sessions, so they are less likely to present spontaneous arguments.

Regardless of such explanations, indications are that certain techniques are more appropriate for some learning activities than others. As the technology for CMC improves, integrated synchronous and asynchronous features will certainly become available. Educators who are responsible for research and development, design, implementation, and evaluation of instruction via CMC will then be faced with the problem of which mode to use for various learning activities. It is hoped that this study has shed some light on that problem.

References

Bales, R.F. (1950). *Interaction process analysis: A method for the study of small groups*. Reading, Mass.: Addison-Wesley Publishing Co.

Beckwith, D. (1987). Group problem-solving via computer conferencing: the realizable potential. *Canadian Journal of Educational Communication, 16*(2), 89-106.

Danziger, K. (1976). *Interpersonal communication.* England: Pergamon Press Inc.

Davie, L.E. (1987). Facilitation of adult learning through computer conferencing. In *The Second Guelph Symposium on Computer Conferencing—Proceedings.* (pp. 11-12). Guelph: University of Guelph.

Gordon, M. (1982). *Nursing diagnosis: Process and Application.* New York: McGraw-Hill Book Co.

Harasim, L. (1989). On-line education: A new domain. In Mason, R. and Kaye, T. (Eds.), *Mindweave: Communication, Computers and Distance Education.* (pp. 50-62). U.K.: Pergamon Press.

Harasim, L. (1987b). Computer-mediated cooperation in education: Group learning networks. In *The Second Guelph Symposium on Computer Conferencing—Proceedings.* (pp. 171-186). Guelph: University of Guelph.

Hiltz, S.R. (1986). The 'virtual classroom': Using computer-mediated communication for university teaching. *Journal of Communication, 36*(2), 95-104.

Hiltz, S.R. and Turoff, M. (1978). *The network nation: Human communication via computer.* Massachusetts: Addison-Wesley Publishing Co., Inc.

Hirokawa, R.Y. (1980). A comparative analysis of communication patterns within effective and ineffective decision-making groups. *Communication Monographs, 47,* 312-321.

Iyer, P.W., Taptich, B.J., and Bernocchi-Losey, D., (1986). *Nursing process and nursing diagnosis.* Philadelphia: W.B. Saunders Company

Johnson, D.W. (1981). Student-student interaction: the neglected variable in education. *Educational Researcher, 10*(1), 1-10.

Johnson, D.W. and Johnson, R.T. (1985). The internal dynamics of cooperative learning groups. In Slavin, R.E. (Ed.), *Learning to cooperate, cooperating to learn* (pp. 103-124). New York: Plenum Press. From the 2nd Conference of the International Association of Cooperation in Education, 1982.

Kerr, E.B., and Hiltz, S.R. (1982). *Computer-mediated communication systems: Status and evaluation.* New York: Academic Press.

Levinson, P. (1988). *Mind at large: Knowing in the technological age.* Connecticut: JAI Press Inc., A series in research in philosophy and technology, supplement 2.

McCreary, E.K. and Van Duran, J. (1987). Educational applications of computer conferencing. *Canadian Journal of Educational Communication, 16*(2), 107-115.

Phillips, G.M., Santoro, G.M., and Kuehn, S.A. (1988). The use of computer-mediated communication in training students in group problem-solving and decision-making techniques. *The American Journal of Distance Education,* 2(1), 38-51.

Powell, J.P. (1986). Small group teaching methods in higher education. In Bligh, D.A. (Ed.), *Teach thinking by discussion* (pp. 26-36). U.K.: Srhe and Nfer-Nelson.

Rice, R.E. and Rogers, E.M., (1984). New methods and data for the study of new media. In Rice, R.E. (Ed.), *The new media: Communication, research, and technology.* (pp. 81-99). California: Sage Publications.

Rogers, L.E. and Farace, R.V. (1975). Analysis of relational communication in dyads: New measurement techniques. *Human Communication Research, 1,* 222-239.

Sleightholm Cairns, B. and O'Kelly, P. (1988). CO-CO in nursing care planning: An innovative approach to student learning. *Sixth Canadian Symposium on Instructional Technology—Proceedings.* (pp. 295-299). Halifax: National Research Council of Canada

Wilton, J.A. (1987). *Evaluation of the software package, CO-CO.* Toronto: Unpublished report. The Peel Board of Education.

Yoder, M.E. (1984). *Introduction to nursing diagnosis* [computer program]. Philadelphia: Lippincott.

Interactive Multimedia Technology and the Community College: Partners in a Twenty-First Century Revolution

Angeline Godwin Dvorak
Enterprise State Jr. College

Abstract

The revolutionary impact of the community college movement highlights a memorable era in the history of American education. As the twenty-first century embarks, the community college must again become a partisan in a revolution. The two-year institution must explore advanced technology to address the educational needs of the next century's generation of learners in an information-centered society. Interactive multimedia technology has the potential to serve as the catalyst for expanding the role and mission of the community college while transforming how learners learn and how teachers teach.

This presentation will feature Enterprise State Junior College as a two-year institution which has assessed the pressing needs of the institution, the community, and business and industry. Consequently, the College has recognized the advantages of using interactive multimedia technology to address problems and deficiencies that the institution cannot presently meet with current personnel or funding. The establishment of a Center for Innovations in Instructional Technology hallmarks Enterprise State's efforts to procreate its vision and to secure educational opportunities for the learners that it serves. The presentation will be supplemented by a videotape which surveys the research and development facitilies, interactive classroom activities as well as the interactive courseware programs currently under development.

The revolutionary impact of the community college highlights a memorable era in the history of American education. The conception of low cost, easily accessible postsecondary instruction birthed a staggering number of two-year, community-based institutions, almost one a day during the peak of its growth. As the twenty-first century embarks, the community college must again become a partisan in a revolution. But this time its goal will not be to establish campuses, staff faculties and recruit non-traditional college students. Rather, the community college will be probing new technologies and exploring innovative paths to identify and address the educational needs of a twenty-first century generation of learners in a technology-oriented society. To realize fully its opportunity to promote educational advancement and reform, the two-year school must redefine its traditional mission as a teaching institution to include a new role as a community-based resource center for research and development in interactive multimedia technology. Synthesizing old and new teaching tools and strategies through the integration of text, computer graphics, audio, still frames and moving pictures, interactive multimedia has the potential to serve as the catalyst for expanding the role and goals of the community college while transforming how teachers teach and how learners learn.

Interactive multimedia technology provides the tool necessary to design, develop and deliver effective stimulating instruction. It empowers the classroom teacher; it empowers the student. When used to prepare transfer students for upper-level studies, to provide elementary and secondary teachers with professional development opportunities, to train the business and industry workforce, and to author and test instructional courseware in a variety of disciplines, interactive multimedia enables the community college to overcome existing obstacles and to meet future challenges.

The next decade poses several problems for the two-year institution, perhaps higher education as a whole, such as teacher shortages and reduced funding, which, in turn, foster increased class size and teaching workloads, limited course offerings and deficient library resources. Beyond serving the needs of its already diverse student population, the community college is usually the most accessible to K-12 teachers as well as business and industry workers. Essentially, society continues to demand that community colleges do more for more with less. In devising a strategy to address these concerns, administrators and teachers must turn to advanced technology to offset the decline of both personnel and funding sources (Phillipo, 1988, p. 62). Educational leaders with vision, in fact, have already begun to employ this new learning tool to prepare students for a technology-based, information-centered society.

Nestled in a rural farming community in southeast Alabama, Enterprise State Junior College has embraced the challenge and is preparing for the revolution. We have realized that we can no longer rely on present strategies or existing resources to meet the needs of the next generation and a new century. Our approach to these challenges centers on the establishment of a Center for Innovations in Instructional Technology. The management plan for designing, structuring and funding a research and resource center, however, is much more important than the physical facility itself. This project involves setting both long-term and short-term goals, tapping all available resources, clustering funding efforts, recruiting resourceful participants, establishing linkages with the community and business and industry, and evaluating each aspect of the project.

Efficient and successful planning begins with analysis. For our institution, the first step involved teaming up innovative, energetic faculty members who are committed to the exploration of educational technology as well as the community college concept. Nothing is more vital to the success of an interdisciplinary team than a leader who is ultimately responsible for project organization and implementation. Careful scheduling and consistent teamwork are essential for rarely are regular teaching duties modified to accommodate the added responsibilities of such a project.

After we conducted our own assessment of what education in the United States needs, what our institution needs and what our specific students need, we went to the students themselves as well as the community and business and industry. The needs analysis revealed several glaring deficiencies in our mission and our curriculum, specifically within the institution, the areas of foreign language studies, courses emphasizing critical thinking and analytical skills, and laboratory classes; in the community, math/science and language arts courseware in K-12 (particularly middle school level); and in business and industry, workplace literacy and basic skills competency. Clearly, the needs are complex and diverse; they reach far beyond traditional resources or funding.

Deciding where to start or which area to address first requires careful evaluation; however, the starting point often simply surfaces as the team solicits the participation of content-area specialists (faculty and non-faculty) and explores opportunities to integrate educational technology into existing curricula or programs under development. Even when the plan is set, the area is identified, and the project is underway, the sudden availability of federal, state or private agency may necessitate a restructuring of priorities and short-term goals. Recognizing interactive multimedia technology as the most efficient, cost-effective tool for addressing these needs, our team concentrated on funding sources that not only specified the target subject area but also encouraged the use of technology for instruction.

For Enterprise State the funding was initially awarded through a Title III grant in which we proposed the development of a first and second year foriegn language curriculum in French, Spanish and German, an English as a Second Language program and the establishment of an interactive multimedia language laboratory. This facility houses twenty student workstations and one instructor control station. Each station is equipped with an IBM InfoWindow System. The systems are then networked with the Tech Commander. This project has served as the foundation as well as the springboard for other project proposals. The hardware, the training, the authoring system and other supports have been used to implement other endeavors while simultaneously serving the original project.

Our efforts have been fueled by a vision of what American education should be and what the community college could be for the twenty-first-century generation of learners. This dream begins with a centralized facility to design and development interactive multimedia courseware, to recruit faculty and staff as innovators in educational technology, to train public and private school educators as well as the American workforce, to implement curriculum integration, to conduct research in the development and evaluation of interactive multimedia instruction, to disseminate research and performance data findings, and to secure funding for all of these efforts.

Having a strategy for the development and funding of a Center for Innovations in Instructional Technology provides a distinct advantage when federal grants or state endowments come available, often on short notice. A proposal is customized to a specific area of the overall design. Our objective at Enterprise State is to prepare ourselves to meet student and community needs. Therefore, we are presently mapping out where we want to go and what is necessary for us to get there. Funding equipment and personnel for curriculum development and integration is a good starting point.

As Karen Scheingold, director of the Center for Technology in Education, has suggested, the only way for technology to make a significant impact on education is to "deeply and purposefully" integrate it into the curriculum and "into what happens in the classroom" (O'Malley, 1989, p. 119). Updating and "overhauling" the curriculum means changing the roles that teachers and students play in the learning process. The progression of technology's role in the educational schema fosters a situation in which both "the responsibility and the pleasure of learning is in the hands of the learner" (O'Malley, 1989, p. 120).

Interactive multimedia technology empowers the community college in much the same way that it empowers the learner. It transforms and expands the dimension and scope of the learning experience and the learning environment. A good interactive application is able to approximate the talents of the gifted teacher and the immediacy of 'hands-on' exposure and to turn students into proactive learners who are free to learn. The use of interactive multimedia technology liberates the two-year institution from the personnel and resource shortages that blight higher education. It enables the community college to fulfill its fundamental mission: to be what any good college should be, "with a qualification: that of access" (Benke, 1989, p. 2).

Certainly, access if the tie that binds. The marriage of the three major communication technologies—print, audio and video—and the personal computer stirs educators' excitement and forces them to see beyond the immediate, the familiar, the convenient. An interactive system allows the student to become the manager-controller of a large body of information that is easily accessible. The student is, therefore, "no longer a mere passive observer of information but rather an active participant" (Gay and Raffensperger, 1989, p. 24). The educator then is the monitor, the facilitator and the author not simply the dispenser of knowledge.

Likewise, with the flip of a switch, a plain, block-walled classroom becomes the most sophisticated chemistry, physics, biology, or foreign language laboratory, the most comprehensive learning resource and research center, the most versatile developmental studies facility available to students in the mainstream of American education.

Paul Benke (1989), president of Jamestown Community College in New York, recently noted:

> Those of us who are part of the community college movement should take a position that we have a responsibility to our founding fathers, not just to our local industries. If there is to be a new America, our community colleges will be one of the major instruments of change. (p. 2)

I second Paul Benke's words and add that those of us who are part of the community college movement should take a position that we have a responsibility to future generations of learners, not just the traditional "college student." If there is to be an innovative, revitalized system of American education, the community colleges which embrace interactive multimedia technology and the advancements that accompany it will be the major instruments of this revolution.

Bibliography

Benke, Paul A. (1989, October 24). Changing Public Perception of Community College. *The Community College Times*, pp. 1-2.

Gay, Geri, & Raffensperger, Ed. (1989, September). Consideration and Strategies in the Design of Interactive Multimedia Programs. *Academic Computing*, pp. 24-25, 57-58.

O'Malley, Christopher. (1989, October). The Revolution is Yet to Come. *Personal Computing*, pp. 115-20.

Phillipo, John. (1988, May/June). A Unique Synthesis of Traditional Teaching Tools. *Electronic Learning*, pp. 60-62.

Adult Self-Directed Learning, Personal Computer Competency, and Learning Style

Helen Barrett
Staff Development Coordinator
Fairbanks North Star Borough School District
P.O. Box 1250, Fairbanks, Alaska 99709
(907) 452-2000 ext. 206

Abstract

In the past decade, many adults, especially in education, have undertaken the task of learning to use a personal computer for their work. Some people have struggled with that effort, while others have found the task to be one of the most productive learning projects of their adult lives. There is also evidence that adults approach a learning project with their own individual learning style. Past research projects have been conducted with students in organized learning environments; however, there is very little research on the efforts of adults to teach themselves how to use this rapidly expanding technology.

In 1989, a research project was conducted to assess the impact of learning style, readiness for self-directed learning and the type of operating system interface on the acquisition of personal computer competency. Approximately half of the participants in the study were from throughout the nation (from an external graduate degree program and attendees at the 1989 NECC conference) and the other half were educators and adult computer applications students from the state of Alaska.

Over 150 participants filled out four instruments: the Kolb Learning Style Instrument (LSI); the Guglielmino Self-Directed Learning Readiness Scale (SDLRS); a Personal Computer Competency Inventory (PCCI); and a questionnaire which asked a variety of questions about strategies for learning to use a personal computer. A smaller number (25) completed a series of qualitative responses to more in-depth questions about their learning experiences.

The data was analyzed by learning style, by type of computer interface preferred, and by level of personal computer competency. A preliminary review of the data suggests that people with one learning style may have more difficulty in gaining personal computer competency (Divergers, favoring concrete experience and reflective observation). Another finding was that, consistent with the literature on adult learning projects, the competent users in the study favored self-directed learning strategies more than 75% of the time.

Background

The personal computer has become a very dynamic factor in recent technological change. In the 1980s, the cost of personal computers has fallen and their power has increased, providing an opportunity for adults to have access to these new tools in their homes as well as in the workplace. This increase in the use of personal computers has also placed new demands on adults as learners, to discover how to make the best use of this technology.

Adults have employed a variety of strategies to gain competence in using personal computers with varying levels of success: classes, user groups, CAI software, books and magazines, consultants and self-directed learning (Ludden, 1985; LaPlante, 1986). Self-education appears to be the most used strategy for adults to use in gaining competence in using a personal computer. However, this method is not without its frustrations, especially if the learner has no prior experience to relate to this new knowledge (Zemke, 1985). The manuals that come with the hardware and/or software are the primary resource used in self-directed learning, along with a problem-solving, trial and error approach. These manuals, however, may not support the adult learning process if they do not address the needs of the learner (Knowles, 1983; Gerver, 1984), or individual learning styles (Galagan, 1987). There is a need to study the self-directed learning processes that adults use to gain competence in using a personal computer.

Most of the research that has been conducted about adults learning to use their computers has been within the context of organized classes or institutional learning. The adult learners in these studies were usually enrolled in some type of organized computer class with an instructor. One very effective method of learning how to use computers has been through cooperative models, such as computer user groups. Computer user groups, because their design adheres more closely to andragogical principles, provide opportunities for learning that are not usually provided by educational institutions. In a recent doctoral research project on adult learning and computers, Ludden (1985) surveyed 91 members of the Northeast Indiana IBM-PC Club. One of the questions on the survey asked what sources were used to learn about computers. 98% of the respondents used "self education" as one source of learning about their computer, with 45% selecting that choice as their primary source of learning (ranked first of all choices). The second choice in the overall (81%) and primary (19%) source of learning were books and magazines (ranked second of all choices). The third ranking choice for primary source of learning (9% each) was a tie between college classes and user group meetings. The subject of Ludden's study, computer use groups, ranked third at 63% of all methods used to learn about computers. These statistics are comparable to other studies of adult learning projects (Penland, 1977; Tough, 1971) where 80% of adult learning projects are planned by an "amateur" (Gross, Tough & Hebert, 1977).

The Touche Ross Enterprise Group surveyed 526 companies with sales between $1 million and $75 million (LaPlante, *InfoWorld*, 1986). "The leading cause of dissatisfaction with microcomputers in smaller and emerging corporations is the amount of time required for training..." (p.31) although it was also noted that "a majority of the respondents—76 percent—say they taught themselves how to operate microcomputers"(*Ibid.*). 11% used a consultant and 9% took a course to learn how to use their computers. These statistics are also comparable to the other research on adult learning mentioned above.

Recently, there have been changes in computer technology which may contribute to more successful learning experiences. Computer hardware has become more sophisticated, resulting in more complex operating systems which allow a more "transparent" user interface. Software has become easier to use and software companies have developed much more readable program documentation. There is also a large amount of materials available to assist the learner, from audio or video tutorials to printed materials (books, periodicals, training aids). The emerging "graphical user interface" (GUI), as contrasted with the "command-line interface," or "traditional/text user interface" (TUI) is also a major focus in the current computer trade press. Recent research sponsored by Apple Computer, Inc., and conducted by Diagnostic Research, Inc. (1988) have supported significant advantages the the GUI, as implemented with the Macintosh, in a number of key areas: user productivity, training time and costs, support time and costs, output, software and personal fulfillment.

Computers and other electronic media are now accepted as performing an increasingly important role in the learning process at all age levels. However, the key variable in implementing these new learning technologies is the willingness to learn and to change established patterns, for both teachers in the classroom and adults pursuing their own learning projects. Becoming competent with a personal computer is placing adults back into intense learning situations which creates a need to understand both the adult learner and the processes of learning to make these activities more effective.

Learning Style Research

The study of learning style is very complex, involving cognition, conceptualization, affect and behavior. (Guild & Garger, 1985). There are basic patterns in personality which "influence many aspects of personal and professional behavior. In general they are called personality styles. When they affect learning, we refer to learning styles" (p.3). Carl Jung (1921) presented one of the first major theories of both individual difference in human psychology as well as "typical differences" (quoted in Guild & Garger,1985), and his work on personality styles forms much of the foundation for the current learning style theory.

There are several different learning style inventories and approaches to the concept of individual differences. For this particular study, I selected the Kolb LSI because if reflects both the cognitive and conceptual components of learning style *and* it is based on an experiential learning model.

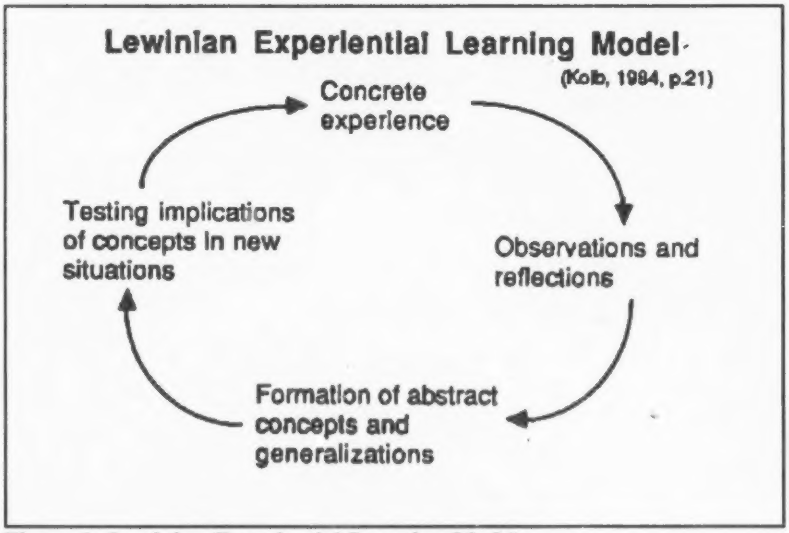

Lewinian Experiential Learning Model
(Kolb, 1984, p.21)
Concrete experience
Observations and reflections
Formation of abstract concepts and generalizations
Testing implications of concepts in new situations

Figure 1. Lewinian Experiential Learning Model

David Kolb (1984) developed a Learning Style Inventory (LSI) which measures two major differences in the way people learn: how they perceive or grasp experience and information (concrete vs. abstract; sensing/feeling vs. thinking); and how they process or transform experience and information (active vs. reflective; doing vs. watching).

The LSI was designed to measure learning styles as a predictor of behavior consistent with experiential learning theory. The latest version of the LSI (1981, revised in 1985) consists of twelve sentences to complete with four phrases to be rank-ordered with numbers from

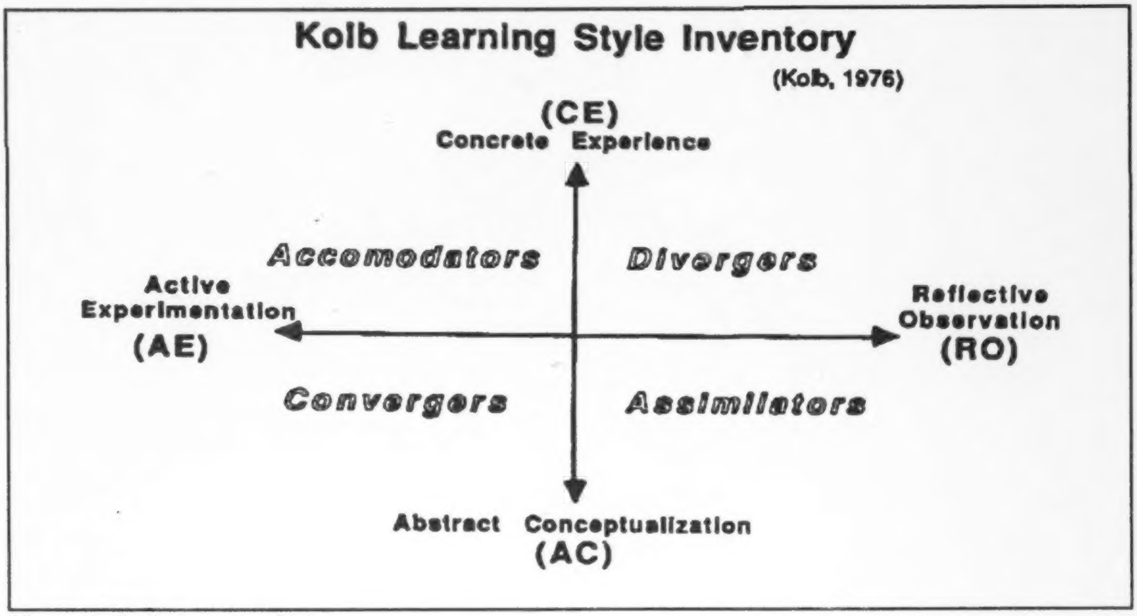


Figure 2. Kolb Learning Style

1 to 4. Each of the choices corresponds to one of the four learning modes. The LSI is a self-reporting instrument that is both *normative*, allowing comparisons between individuals; and *ipsative*, where the "strength of each learning style category is expressed, not in absolute terms, but in relation to the strength of the respondent's other learning style preference" (Sewall, 1986, p.56).

Research Purpose

There has been a need for an in-depth study of adult, self directed learning, particularly of how adults learn to use personal computers, which will result in better techniques and more understanding of the process of teaching oneself how to use them as a productivity tool. As more adults become involved in the process of selecting and using personal computers for learning, working and problem-solving, there will be an increased need for knowledge about the successful learning strategies of competent computer users. There have been no major research projects which describe the self-directed learning process of gaining competence in learning to use a personal computer. It will be shown that, with a large majority of adults using this method to learn how to use their personal computers, knowledge about this process would contribute to both the theory and practice of self-directed learning in the information age.

The goals of this research project were:

To explore the role of learning style as a factor in the process of learning to use a personal computer.

To explore the role of that readiness for self-directed learning has on the acquisition of personal computer competency

To develop an understanding of the impact that the type of computer operating system (graphical vs text user interface) has on the learning process of learners with different learning styles

Design

This descriptive research project used four different quantitative instruments: Kolb's Learning Style Inventory (LSI); Guglielmino's Self-Directed Learning Readiness Scale (SDLRS); a Personal Computer Competency Inventory (PCCI), developed by the author and another doctoral student in Anchorage, Alaska; and a questionnaire developed by the author which inquired about computer learning strategies, personal computer ownership and use. In addition, there were a series of qualitative responses to some open-ended questions based on Raymond Wlodkowski's (1985) "Time Continuum Model of Motivation" to inquire more deeply into different phases of the learning process.

As of the end of September, 1989, 155 people have responded to all of the questionnaires, with 25 of them completing the optional essay questions. Participation was solicited from several sources: educators attending the Alaska Association for Computers in Education annual conference in March, 1989; attendees at the NECC annual conference in June, 1989; graduate students and faculty at The Fielding Institute's annual Summer Session, 1989; participants in adult computer classes at the University of Alaska Fairbanks, School of

Career and Continuing Education, in the fall of 1989; and employees of the Fairbanks North Star Borough School District.

Efforts were made to find a cross section of users in four categories, based on each participants perceived level of their own competence (beginning or competent user) and their preference for either the graphical user interface (GUI) or the text/command-line user interface. This resulted in four main groups, with percentages noted as of September, 1989:

BT - Beginning users preferring *text* user interface operating systems 11% - n=17

BG - Beginning users preferring *graphical* user interface operating systems 8% - n=13

CT - Competent users preferring *text* user interface operating systems 32% - n=50

CG - Competent users preferring *graphical* user interface operating systems 48% - n=75

The learning styles of the participants were:

Divergers - *Concrete Experience & Reflective Observation* 18% - n=27

Assimilators - *Abstract Conceptualization & Reflective Observation* 27% - n=42

Convergers - *Abstract Conceptualization & Active Experimentation* 30% - n=46

Accomodators - *Concrete Experience & Active Experimentation* 25% - n=40

Preliminary Findings

As this paper is being written, data is still being gathered and statistical analysis procedures just being developed. Greater effort is being made to gather more information from beginning computer users (who are becoming harder to find as the computer revolution matures). Therefore, this paper will cover the preliminary data available at the end of September, 1989. A complete report will be available at the NECC conference in June, 1990.

This was not a random sample of computer users, since an effort was made to find a cross-section of users at different stages of the learning process. No effort will be made to generalize to a larger population. One interesting preliminary observation about the participants in this study is the slight preference for the Active Experimentation (AE) learning style over the Reflective Observation (RO), especially of the Competent users. Active users were 54% of the total group, but 59% of the competent users (n=125) and only 33% or the beginning users (n=30). In an analysis of the items on Kolb's instrument, the characteristics of the active learner (active, practical, hard work, results-oriented, try things out and practice) are favored practices in learning a personal computer according to the responses to the questionnaire. The characteristics of a reflective learner (watching and listening, observing, carefulness, taking time before acting) do not appear to be strategies favored by the successful learners.

Perhaps another reason for the higher percentage of AE learners may be that those people in this study adopted the emerging personal computer technology earlier than the RO learners. The average first year of personal computer use was 1982 for the competent users and 1985 for beginning users. The average number of years of active computer use was 5.8 for the competent user and 2.4 for the beginning user. The average number of years for the AE user was 6.2, and 5.4 for the RO user.

There was also a preference for the Abstract Conceptualization (AC) learning style (56%, n=87) over the Concrete Experience (CE) learning style (44%, n=67). There was a leaning toward the AC style in those who preferred the graphical user interface (63%, n=55) as opposed to the CE style (37%, n=33). Those who preferred the text user interface were evenly split between both these styles (AC= 48%, n=32; CE=52%, n=34). Kolb's description of the abstract learner (logical thinking, reasoning, ideas and theories, analysis), fits the common stereotypical description of a computer system, whereas the description of the concrete (sensory) learner (feelings, intuition, personal involvement, relationships) are characteristics that even developers of "artificial intelligence" concede are not within our current computer capability (Dreyfus & Dreyfus, 1986).

In addition to learning style, a variety of questions were asked relating to learning a personal computer. In looking at the reasons that people wanted to learn a PC, personal curiosity (84%), school work/learning tool (77%), save time and effort (70%) and general career advancement (60%) headed the list. In responding to *who* helped learn in both the early stages and current learning, the overwhelming preference was for "myself" (78% early and 81% now). When asked the *single* most important source of human assistance, the reliance on the self outscored all other responses (42% early and 57% now). In another question regarding the percentage of time spent in self-directed learning vs. organized learning activities, the competent users indicated an average in excess of 75% of the time was spent in self-directed learning (the beginners averaged 62%).

Another question was asked about all sources of assistance in learning to use a personal computer. The following table reflects the responses of the participants. Again, "hands-on experimentation," a self-directed learning activity, followed by software manuals (another support tool for self-teaching), were the overwhelming choice of the participants.

All Sources Used		Primary Source
55%	Classes for college credit	9%
55%	Non-credit workshops or short courses	3%
39%	Conferences	2%
25%	User group meetings	0%
68%	Software tutorials on the computer	1%
68%	Books and magazines	6%
94%	Hands-on experimenting with the computer	47%
86%	Software Manuals	17%
61%	Family & Friends	10%
12%	Television shows about computers	0%
15%	Audio or Video Training Tapes	0%

It should be noted that there was a tremendous difference between the beginning and competent users' responses to this question. The competent users have used many more learning opportunities. The chart below shows *all* sources used by each group.

Beginners		Competent
43%	Classes for college credit	58%
40%	Non-credit workshops or short courses	58%
10%	Conferences	46%
13%	User group meetings	27%
70%	Software tutorials on the computer	68%
40%	Books and magazines	75%
97%	Hands-on experimenting with the computer	93%
67%	Software Manuals	91%
70%	Family & Friends	59%
3%	Television shows about computers	14%
10%	Audio or Video Training Tapes	16%

To find out if prior experience had any impact on the level of personal computer competency (on a scale of 1 [no experience] to 5 [a lot]) there was little experience with mini or mainframe computers (M=1.8), memory typewriters (M=1.7) or dedicated word processors (M=1.5). Most people had more experience with typewriters (M=4.1) and hand calculators (M=3.9) than any other type of computing device.

Questions were asked about the problems encountered during the learning process. On a scale of 1 (never a problem) to 5 (very frequently a problem), the following is a summary of the responses. The mean scores for the beginners were a full .5 points higher on the mean scores marked with an asterisk (*) below. Note the high scores on the last item (Never enough time!!).

	Mean	*S.D.*
Not enough time because of other responsibilities	3.72 *	1.16
Manual hard to read	3.48 *	1.16
Resources too expensive	3.05	1.32
Trouble getting help	2.81 *	1.20
Difficulty with software commands	2.80 *	1.14
Resources not available	2.71	1.23
Lack of available courses/workshops	2.64	1.30
Problems with computer hardware	2.57	.91
Difficulty with operating system	2.55 *	1.05
Experts were not knowledgeable	2.42	1.16

When asked to indicate how they preferred to learn, 42% said they preferred to learn alone, 21% in a small group, 22% with a friend or relative. Only 8% preferred to learn in a formal class structure and 7% preferred some other method or combination of the above. Only 23% were members of a computer user group, although 40% had attended a user group meeting.

It also appears that the manuals that come with software are not fully used, that a majority of the participants never read the manuals, perhaps skimming them to see what the program does, and consulting the manual only when they have a problem they can't solve. When asked what they like to do *first* when learning a new program, 34% (n=52) responded, "Turn on the computer and start experimenting." 21% (n=32) liked to work through a tutorial disk, if it was available. 16% (n=25) wanted to watch a demonstration and only 13% (n=20) would read the manual first. Some of the respondents to the essay questions suggested that software manual writers do much more "beta testing" with novice users, to avoid the assumptions of prior knowledge that are often the weaknesses of many manuals.

Computer ownership appears to be an important factor in gaining computer competency. While only 10% of the participants in the study did not own a computer, 20% of the beginners were non-owners, compared to 7% of the competent users. The competent users also used the assistance from colleagues at work, computer dealers and user group members than the beginners in this study. When asked where they learned their first computer system, only 20% of the beginners selected work, compared to 38% of the competent users, which may indicate the importance of organizational support for introducing and adopting new innovations.

One interesting finding was that there did not appear to be any difference in readiness for self-directed learning, as measured by the SDLRS, by learning style, level of competency or type of operating system. The mean scores of all groups were within 4 points of the mean score of 246, which falls between the 83rd and 88th percentile on Guglielmino's rating scale.

Conclusions

Regardless of the expense and effort devoted to organized computer training program, self-directed learning is the primary method that most adults employ to gain competence in using a personal computer. There is no substitute for the "discovery learning," inductive, experiential approach that most adults appear to use, based on the strategies they prefer. Learning style appears to have an impact of the learning strategies used as well as the self-reported level of personal computer competency for the people in this study. Learners who preferred the reflective observation (RO) and concrete experience (CE) learning styles expressed more difficulty in learning and had significantly lower personal computer competency scores based on the self-report inventory used.

The qualitative data, not reported here, provides a rich source of information about learning different operating systems, further supporting the "ease-of-use" characteristics of the graphical user interface. A few of the participants reported significant life changes as a result of their learning activities, including greater self-awareness, self-confidence and career advancement.

Bibliography

Brookfield, Stephen D. (1985). *Self-Directed Learning: From Theory to Practice*. San Francisco: Jossey-Bass.

Cross, K. Patricia. (1981). *Adults as Learners*. San Francisco: Jossey-Bass Inc.

Diagnostic Research, Inc. (1988) *Macintosh or MS-DOS?* research report sponsored by Apple Computer Co., Inc.

Dreyfus, Hubert L. & Dreyfus, Stuart E. (1986) *Mind Over Machine*. New York: The Free Press.

Galagan, Patricia. (1987) "Computers and Training: Allies of Enemies?" *Training & Development Journal. 41*, 4, pp.73-76.

Gerver, Elisabeth. (1984). *Computers and Adult Learning*. Milton Keynes: Open University Press.

Gross, Ron; Tough, Allen; Hebert, Tom. (1977) "Independent, Self-Directed Learners in American Life: The Other 80% of Learning" in *Yearbook of Adult & Continuing Education*. Chicago: Marquis Academic Media. 4th Ed. 1978-79, pp. 43-77.

Guglielmino, Lucy M. (1977) "Development of the Self-Directed Learning Readiness Scale" (Doctoral dissertation, University of Georgia)

Guild, Patricia, and Garger, Stephen. (1985) *Marching to Different Drummers*. ASCD.

Knowles, Malcolm S. (1983). Malcolm Knowles finds a worm in his Apple. *Training and Development Journal. 37*(5), 12-15.

Kolb, David A. (1984) *Experiential Learning*. Englewood Cliffs: Prentice-Hall.

Kolb, David A. (1985) *Learning Style Inventory*. Boston: McBer & Co.

LaPlant, Alice. (1986) "Top PC Complaint of Small Companies is Time Used for Training, Survey Says."in *Infoworld*, October 20, 1986, p.31.

Ludden, Laverne Lee. (1985) "Grassroots Computer Education: A Study of Computer User Groups and the Model they provide for learning to cope with new technology." (Doctoral dissertation, Ball State University)

Mruk, Christopher J. (1984) Facilitating the Acquisition of Computer Skills for Adults: A Handbook of Findings and Recommendations. Washington, D.C.: Office of Human Development Services (DHHS). Paper presented at the National Educational Computing Conference. ERIC # ED 277 338

Penland, Patrick R. (1977) *Self-Planned Learning in America*. Pittsburgh: University of Pittsburgh

Russell, Susan Jo. (1983) "Teachers and Computers: Reflections on Learning" in National Staff Development Council's *Journal of Staff Development*. 4(2), pp. 101-107.

Sewall, Timothy J. (1986) "The Measurement of Learning Style: A Critique of Four Assessment Tools" ERIC # ED 267 247

Tough, Allen. (1971). *The Adult's Learning Projects: A fresh approach to theory and practice in adult learning*. Toronto: Ontario Institute for Studies in Education.

Turkle, Sherry. *The Second Self*.

Wlodkowski, Raymond J. (1985). *Enhancing Adult Motivation to Learn*. San Francisco: Jossey-Bass Inc.

Zemke, Ron. (1985). *Computer-Literacy Needs Assessment*. Reading: Addison-Wesley Publishing Co.

M4-8 ISSUES IN TWO-YEAR COLLEGE COMPUTER EDUCATION (ACM/SIGCSE)

Issues in Two-Year College Computer Education

John Impagliazzo
Hofstra University

Helene Chlopan
University of Kentucky

Herb Garrett
Long Beach City Community College

Margaret Heard
East Central College

Karl Klee
Towson State University

Abstract

Two-year college computer education in the United States is going through a period of transition. Curricula were once classified as either computer science or information systems (data processing). Because of demands imposed by the job market, new areas of computer education, such as computer technology, computer engineering, or program operations, are now emerging. The panel will discuss how the computer needs of a changing society impacts upon the 2-year college educational community. An open discussion will follow the presentation.

Teaching a Multidisciplinary Topic Computer Integrated Manufacturing (CIM)—A Case Study

Lee Danner, Assistant Professor, Computer Science
Allen E. Smith, Assistant Professor, Management & Marketing
George Stanton, Associate Professor, Technology

Computer & Information Science
P. O. Box 23830A
East Tennessee State University
Johnson City, TN 37614-0002
(615)929-6961 Bitnet I01LEE@ETSUACE

Abstract

Integration of manufacturing processes, business processes, and information handling processes have led to improved quality, decreased cost, and added flexibility in manufacturing enterprises. This integration is seen as a necessary condition for American manufacturers to remain competitive in the world marketplace.

East Tennessee State University is developing an undergraduate curriculum in Computer Integrated Manufacturing (CIM) that addresses concepts and techniques used in this area.

The initial course of this curriculum is being taught as an interdisciplinary course involving faculty from Management & Marketing, Manufacturing Engineering Technology, and Computer Sciences. The interdisciplinary teaching parallels the integrated approach taken in implementing CIM concepts in industry.

Some advantages and shortcomings in this approach are presented. Updated results will also be made available at the conference.

Teaching integration as a multidisciplinary course integrates the teaching environment while presenting an integrated topic.

Introduction

East Tennessee State University, Johnson City, TN, is offering a course (first taught in the fall semester of 1989) entitled "Introduction to Computer Integrated Manufacturing Concepts". The course was developed jointly by faculty from the School of Business, Management & Marketing department and The School of Applied Science & Technology, Technology and Computer & Information Sciences departments. Lectures, assignments, discussions, and field trips all involve the faculty representatives from these three areas and a truly integrated course is being taught on an integrated subject.

Importance of CIM

Since the end of World War II, the American economy has undergone significant change--many years of boom followed by years of erosion. Currently a large percentage of consumer goods intended for domestic consumption are produced overseas. This situation has caused a drain on the American economy with a significant trade deficits and a large debt to foreign nations. Success at reversing this rising trade deficit lies in improving the competitiveness of our domestic products.

The magnitude of the problem can be seen by recognizing that manufacturing comprises approximately one fifth of our gross national product and that any improvement in the manufacturing process will result in a significant effect on improving the economic situation. The two principal avenues of improvement are through better management and through technology [Green88].

The gains to be achieved are: the ability to support a drastically shortened product life cycle, to meet the higher quality expectations of consumers, to improve the national economy, to take advantage of rapidly developing technologies, and to relieve the intense pressure being applied by foreign manufacturers [Fry89]

The gains accomplish the overall objectives of improved products, decreased costs, and additional manufacturing flexibility.

Manufacturing flexibility is an important accomplishment in itself, since it is expected that in the next few years, 75% of all manufactured parts will be in batches of less than 50 units [Groo80]. With this requirement, a manufacturing process that can adapt to design and product change rapidly is a necessity. The manufacturer who can produce more customized products, at less cost, less lead time, and provide higher quality, has a distinct advantage over those who cannot. The job shop that produced individual quality products in the past was often replaced by production lines, paced by machines. This job shop, coupled with the automation available through the technology of information systems, comprising a Flexible

Manufacturing System (FMS) is once again a desirable entity and is part of CIM.

Introduction to CIM

The name Computer Integrated Manufacturing is both a fortunate choice and an unfortunate choice of a name to describe the automation of a manufacturing company. The fortunate part is the word "integrated". In CIM, several disciplines are truly integrated into the manufacturing process, including using the computer to assist in engineering design (Computer Automated Design), using the computer to assist in the set-up and operation of the automated manufacturing equipment (Computer Automated Manufacturing), and optimizing the use of the manufacturing resources (Computer Integrated Manufacturing in its early interpretations [Goet88]).

The unfortunate choice of the label is that CIM is not restricted merely to manufacturing. It is an operating philosophy that aims at greater efficiencies across the whole production cycle, including product design, manufacture, and marketing [Green89]. Furthermore the term is not limited just to technology in the production cycle. The full scope of CIM is the integration of all of the processes of manufacturing, by using computers to automate the flow of information throughout the operation. This includes the integration of marketing, personnel, finance, and other business applications into the process. With these applications also integrated into the business of manufacturing, we can better accomplish the goal of reducing the cost of manufacturing while increasing its flexibility. Thus we are looking at more than computers integrated into an application (CIM); we are looking at the Computer Integrated Enterprise (Business, Manufacturing, & Computing).

CIM in the Curriculum

There are a number of areas where the multidisciplinary integration in this course presents lessons useful to the students, to the faculty, and to the university. They are:

- Physical separation of the departments
- Syllabus divisions
- Examinations and grades
- Different approaches to solutions
- Different approaches to information presentation
- Buzz-words and acronyms galore.
- Multiple teaching styles
- Incompatible departmental productivity tools
- Isolated departmental computing equipment
- Follow-on courses and curriculum

Physical separation of the departments

The management, technology, and computer science departments are in different buildings on the main campus of ETSU. This separation is a minor obstacle to teaching the course; but it does affect casual communications between the faculty members and individual schedules must include time necessary to walk across campus for class meetings. Although minor, these obstacles parallel those in business enterprises, where the engineering department is usually located separately from manufacturing and the front office is also, as its idiomatic name infers, somewhere else. These physical separations ultimately lead to functional separation of the enterprise into cells of expertise (not unlike the University). Reversal of this separation is the real meaning of integration.

Syllabus divisions

The syllabus for the course is divided to support the three disciplines of the course. Background materials, definitions, and short histories of the development of each of the disciplines are first. This allows comparing and contrasting these topics and the separate view points of each of the faculty. For example, the history of manufacturing, the history of production management, and the history of information processing have all followed very different paths. The time spans represented by these histories are quite dissimilar. The history of manufacturing began with the first agrarian societies in the early history of civilization. The history of business begins much later with the feudal empires of the renaissance, and the history of information processing began far more recently with the invention tabulating machinery, near the end of the 19th century with its real beginnings with the invention and availability of the stored program computer in the 1940s.

The parallel between the punched card looms for manufacturing fabrics and the punched card tabulating equipment used with the 1890 census is but one of the many similarities of these histories. In the business area, meanwhile, we saw the development of Scientific Management using time studies and later time & motion studies in an attempt to improve the productivity of a manufacturing environment or factory.

The background and historical material was followed by the first examination and a field trip to a manufacturing facility where CIM has not yet been implemented. Operations in this facility were evaluated in comparison with the facility described in the early chapters of [Gold86]. The remainder of the classroom lectures are on the technical content of the course and is divided into three hour segments (one week of classes) rotating through the three disciplines.

All three professors attend each of the lectures and occasionally interject their views into the materials being presented. This interaction supports the theme of the material and serves as a catalyst for student participation. Faculty interactions in the lecture must be carefully limited to avoid preempting interaction between the students and the lecturer *du jour*.

At the end of the semester, review showed that a strict three hour rotation was inappropriate and that each area should have sufficient contiguous lectures to cover a topic. We also discovered that we had overestimated the time available in one third of a course and had spent too much time on introductory material and not enough on the principal concepts of CIM.

Examinations and grades

The students enrolled in the course were recruited from majors in each of the departments. Consequently, the course material in the student's major is easier than that from the other disciplines. Examinations cover all three disciplines, with the professor in the discipline responsible for the grading of the questions in his area. These two independent variables make it necessary to have a combined review, prior to returning the grades to the students. This is not an unworkable process, but is another unique situation that requires attention.

The resulting grades were apparently fair (for a rather small sample) and there were no apparent biases in favor of students in one discipline over another.

Different approaches to solutions

The problem solving processes used by the management student, the technology student, and the computer science student will not necessarily be the same. Formal methods are stressed in the computer science curriculum while ad hoc methods are avoided. The other disciplines may or may not include this material. More traditional or less formal methods used in the other curricula are another item that may require compensation in the teaching and administration of the course. (In retrospect, at the end of the course, there was no apparent problem in this area.)

Different approaches to information presentation

The mechanisms used to present information differ between the disciplines. A tool used in one discipline may be completely unheard of in another. Charts and graphs are common to all, but engineering drawings, engineering specifications, decision tables, pseudocode, models, and others may have one significance in one discipline and another in one of the others. Just as world communications are diminished by language differences, jargon and presentation styles may cause communication break-downs in the multidisciplinary approach to teaching.

Awareness of this potential problem and student maturity resulting from taking courses in different departments of the university appear to have been sufficient to overcome these fears.

Buzz-words and acronyms galore

Each of the disciplines (Business, Manufacturing Technology, and Computer Science) has its own language. To paraphrase Winston Churchill, we are three groups of people separated by a common language. Common terms may have a specific meaning in one of the disciplines that is unknown in the others. Acronyms like JIT, CPU, FMS, CNC, NC, etc. are common in modern usage and are used to shorten the length of a communication. When unknown by one of the parties (professors, students, or both), a failure to communicate results. For this reason we endeavored to always define our acronyms before use. Slips did occur, and when questioned by a student, resulted in longer communications paths.

Multiple teaching styles

Different instructors have different teaching styles ranging from intense (fast) lecture to completely interactive, with much student participation. This is another minor difference encountered in the multidisciplinary approach being taken on the subject of CIM, but it does affect the classroom environment.

The presentation styles of the three instructors in this course was markedly different and each transition provided a welcome break, not unlike the appearance of a guest lecturer in a traditional course.

Incompatible departmental productivity tools

Another, not unexpected, obstacle is that course materials for the three parts of the class utilize different word processing and graphics packages. The flow of written materials must flow through some compatibility interface for electronic exchange or must be transmitted by hard copy (a step backward in modern communications).

Isolated departmental computing equipment

Each of the departments involved in the course have departmental computing equipment for their faculty and staff. In the terminology of CIM, these are islands of automation. Within the departments there are also communications facilities (local area networks) to allow intercommunication between members of the department. The long range plan for the university is to interconnect these LANs to provide campus-wide communications. The This innovation is scheduled for near-term implementation and will provide the required communication links. Meanwhile, the campus resembles the pre-CIM manufacturing facility and is dependent on physical transmission of information.

Follow-on courses and curriculum

The interdisciplinary teaching of the introductory course will not be appropriate for more advanced courses in the CIM curriculum. Planned courses will include: Advanced CIM Concepts & Applications, The Computer Integrated Enterprise, Statistical Applications in the CIM Environment, Organizational Behavior in the CIM Environment, and others as deemed appropriate. These prospective courses are all subject to change, but by their nature will each require more depth in one or more of the disciplines of the introductory course (described here).

Conclusions

Integration of manufacturing processes, business processes, and information handling processes have led to improved quality, decreased cost, and added flexibility in manufacturing enterprises. The resulting improvements will improve the competitiveness of the enterprise and when sufficiently integrated into the nations economy, could have significant impact on changing the trade balance in a positive direction.

ETSU is developing a curriculum in CIM that begins with an Introduction to CIM Concepts course. This course is being taught as an interdisciplinary course involving faculty from Management & Marketing, Manufacturing Technology, and Computer Sciences. The interdisciplinary teaching parallels the integrated approach taken in implementing CIM concepts in industry. All three areas have an important contribution to make to CIM and to the course. Maintaining awareness of the problems and benefits of the approach can be used as a positive feedback into the course material. At the time of this writing, the first session of the course is underway. (Some materials have been added at the completion of the first session.) At the time of delivery of this paper, two sessions will have been completed and additional information on the success of the approach will be available and will be presented. Additional problem areas will also have been encountered and will also be described.

The results of this first course will also be used in the development of a complete curriculum in CIM at ETSU.

Bibliography

[Fry89] Fry, Timothy D., and Allen E. Smith, *FMS Implementation Procedure: A Case Study*, IIE Transactions, September 1989.

[Goet88] Goetsch, David L., *Fundamentals of CIM Technology*, Delmar Publishers Inc., Albany NY, 1988.

[Gold86] Glodratt, Eliyahu M. and Jeff Cox, *The Goal, A Process of Ongoing Improvement*, Revised Edition, North River Press Inc., New Haven CT, 1986.

[Gree89] Greenwood, Frank, *Introduction to Computer-Integrated Manufacturing*, Harcourt Brace Jovanovich, Publishers, 1989.

[Groo80] Groover, M. P., *Automation, Production Systems, and Computer Aided Manufacturing*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1980.

87

A Software for Computer-Aided Engineering Education

P.J.Tatsch, F.Damiani and N.Marranghello
Faculty of Electrical Engineering
University of Campinas—UNICAMP
13081 Campinas, SP—BRAZIL

Abstract

This paper describes a user interactive software for electronic engineering education—UISEE—designed to give didactic support to graduation courses or to the updating of graduated students.

With the aid of pop-up menus and windows, the students can choose a subject, get information about it, vary the parameters of the models that represent a given subject and observe its influence on the results obtained, and compare the results of different models with experimental data.

The software is managed by supervising routines which compute and show program usage statistics and allow the interaction between the instructor and the students. An editing facility enables the instructor to create new subjects and modify existing ones.

UISEE is written mainly in C for the sake of portability.

Introduction

The understanding of concepts and models of the behavior of physical phenomena as well as of their application to physical systems is usually taught through conventional exercising[1]. A standard set of exercises cannot, however, span a wide range of parameters, variables and model types. The understanding of the subject is much enhanced whenever the student has the possibility to:

- compare measured data with model results;
- compare results of different models;
- observe model limitations and its range of validity;
- observe the effects of scaling of parameters and variables;
- verify to sensitivity of a model to parameter variations.

A clear, straightforward and quick way of doing so can be implemented in a software environment. In such an environment one can, for instance, change the parameters and variables ranges in a continuous and interacting way and immediately observe the effects of such variations in a graphics display output.

UISEE (User Interactive Software for Engineering Education) has been designed to support all the above characteristics in the field of semiconductor physics and devices. In order to achieve a good portability the C

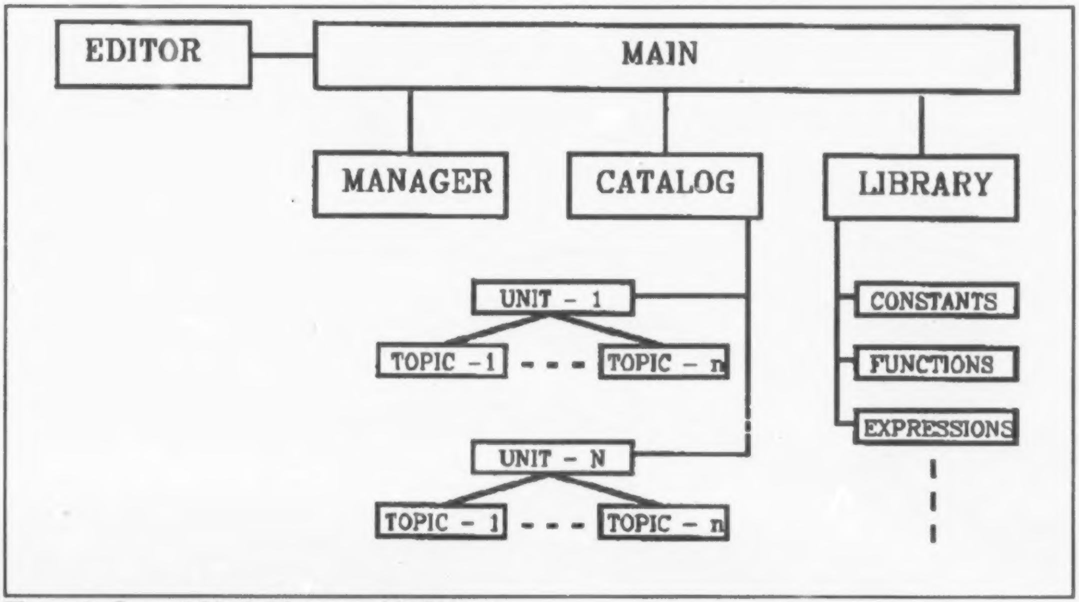


Figure 1. General Block Diagram of UISEE

programming language was chosen for the development of this package, that can run on PC's.

The next sections describe UISEE's structure, its behavior from both the student's and the instructor's points of view, and an example.

UISEE Structure

A general block diagram of UISEE structure is shown in Figure 1. The structure is modular and hierarchical comprising an **editor** program and a **main** program to which a **catalog**, a **library** and a **manager** module are connected.

The **main** program controls the processing of the **manager** and **catalog** module routines and provides error messages whenever necessary. Through the main program a supervisor can also access the **editor**.

The **manager** module performs statistics about **catalog** usage and provides a communication link between the supervisor and the students. Through the use of an expert system, in the future it will be able to directly answer questions and make suggestions to the student about units and topics usage based on the student statistics. The student identification routine (SIR) belongs to this module. The SIR checks the student name against a list of authorized users. Following this it opens the log file of the student and writes the date and time of access. **Units** are the instances of scientific or technological knowledge while the **topics** are particular instances of the corresponding **unit** subject. Every **unit** and **topic** program writes to this log file the times it was inputted and outputted. When the student quits the program the SIR closes the log file. The SIR also writes to the log file the student questions and suggestions, and writes to an existing questions file (EQF) a flag indicating the name of the student and the date of the question or suggestion. When the supervisor enters the **manager** it checks for an entry in the EQF, warning the supervisor if there is any. The supervisor can run an answering routine. A user statistics routine (USR) shows the usage statistics on each particular unit or topic, including question rates, in order to provide means for evaluation and optimization of the package. This will allow the supervisor to tailor the package to the needs of the users, e.g., modifying and/or creating **units** or **topics**.

The **library** module contains physical constants, numeric values files, general mathematical functions, the mathematical expressions and physical descriptions of the models, routines for the graphical treatment of data and other miscellaneous functions.

The **catalog** module is comprised by several **units**. Each **unit** contains the related **topics**. Each **topic** program is an assembling of **library** module functions that perform the **topic** subject.

The **editor** module is defined as a shell built around a C compiler, a debugger and a text editor. It allows the supervisor to access directly all the package and to edit, debug and run any part of it.

User and Supervisor Interfaces

This section describes the high level organization of the program without taking into account its software implementation.

As soon as the main program is run, the user is faced with a student identification routine. After program acknowledgment, an opening screen is presented. This screen contains the list of available **units**, e.g., *MOS transistors*. By choosing **a unit** the user is directed to a second screen containing a set of **topics**. The nature of a topic program can be descriptive, e.g., *MOS structures*, user interactive, e.g., *PN junction*, or both. After choosing **a topic** a third screen appears, containing pop-up menus through which all actions and options can be performed.

Whenever applicable inside a **topic** the student can view in windows the mathematical expression of a model, view and edit the numerical values of parameters, of ranges and of constants. The graphics available in the **topic** can be viewed together or in any number on the screen. There is also a zooming facility. It is possible to view on the same graph a superposition of the output of runs using different parameters for a model or even for the different models available.

A comprehensive help facility is also provided.

A printed hardcopy of the screen contents plus, as an option, the numerical values of constants, ranges of variables, parameters, model types and their mathematical expressions can be obtained.

Having finished to use one **topic**, the student can go back and choose another topic of the same **unit** or go further back and choose another **unit** or quit.

The student can also leave questions and suggestions to the supervisor of the package.

Additions of new models to **topics**, new **topics** to **units** and new **units** to the package, the update of existing modules as well as the establishment of a user list and the control of student performance using the **manager** module routines will be performed by one or more professors/instructors which will be called supervisor. The supervisor will also answer the questions and appreciate the suggestions left by the students that used the package.

To be recognized as a supervisor, after being prompted by the identification routine one has to identify him/herself as supervisor. Then, the identification routine will ask for a password. The reason for this is to maintain the integrity of UISEE.

After the acknowledgment, the supervisor can choose between using the **editor** program, the **manager** module or the **catalog** module.

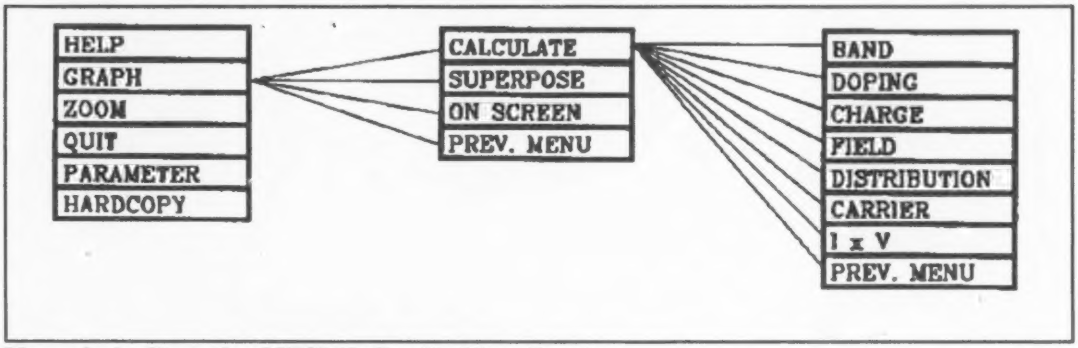


Figure 2. An Example of UISEE's Pop-Up Menu Structure

An Example

Figure 2 shows the pop-up menu structure in the case the Bipolar Devices unit and the PN Silicon Homojunction **topic** are chosen in higher level menus.

Two types of options are available in the menus: interactive and non-interactive. The former type includes options such as *graph*, *model* and *calculate*. To the latter belong options such as *help*, *zoom* and *superpose*, which work like orders to the system. Due to the hierarchical structure of the program each of the interactive options conveys a series of other options that can be chosen in lower level pop-ups.

Besides that, the user can go down the various pop-up levels by choosing some interactive option or go back to higher levels through the use of the *previous menu* option existent in all but the highest pop-up level.

Conclusions

UISEE was designed and partially coded. A great care was taken in the design of the program's architecture. The portability and ease of use as well as the possibility of inclusion and modification of units and topics were of great concern to us. This system can be used as a teaching tool also for other areas of education needing the same kind of interaction with the students.

Presently a few topics of the catalog module have been coded for the sake of package optimization through the evaluation of its performance. Also, the coding of the first version of the manager module is being finished. The coding of the editor module will be done last.The Borland Turbo C version 2.0 is being used for the coding.

Acknowledgments

We would like to thank C.S.Yamaguchi, majoring in Computer Sciences, for the coding of the modules of this package.

References

[1] Damiani, F. and Tatsch, P. J., "Dispositivos Eletrônicos", private publication.

Using Circuitmaker in a Digital Logic Class

Terry A. Scott
Peter C. Isaacson

Computer Science Component
Department of Mathematics and Applied Statistics
University of Northern Colorado
Greeley, CO 80639
(303) 351-2630
compsci@unc.colorado.edu

Abstract

This paper discusses a software program called CircuitMaker, which runs on the Macintosh computer. This program allows the user to design, draw, and simulate digital electronic circuits. This software is being used in the laboratory portion of a Digital Logic class offered at the University of Northern Colorado.

The laboratory portion of Digital Logic classes has traditionally used hardware components and a breadboard to create and test circuits. The advantages and disadvantages of the CircuitMaker software over the traditional hardware approach are discussed.

A brief description of the various menu choices and the operation of the CircuitMaker are given. This includes a discussion of the macro facility and the built in libraries of standard electronic components such as switches, LEDs, and TTL integrated circuits.

In using the software a couple of program bugs have been found. These are described and solutions to these are discussed.

Introduction

Computer Science students at the University of Northern Colorado are expected to take a course in Digital Logic during the Spring Semester of their freshman year. The purpose of this course is to give the students a basic understanding of the electronic hardware used in computers and the workings of a CPU. The course basically consists of a 2 semester credit hour lecture and a 1 semester credit hour laboratory.

The lecture deals with Boolean algebra, combinational circuits, and synchronous circuits. The book used for the lecture portion of this class has been *Digital Design* by M. Morris Mano [Man 1984]. The book used Spring Semester of 1990 was *Fundamentals of Logic Design* by Charles H. Roth, Jr. [Rot 1985]. Both books cover the fundamental concepts and seem to be very good books. The presentation is different in the two books and it was decided to try the Roth book to see if it helped the students' understanding.

The laboratory portion of the class gives the students some hands-on experience in designing and building such things as timers, decoders, multiplexers, adders and control circuits.

Previous to the Spring Semester of 1989, the laboratory portion of the course was taught by using a hardware design box called a Pencilbox Logic Designer manufactured by E. L. Instruments. These boxes were purchased in 1984 for approximately $100 each. Each box contains switches, LED's, batteries, a breadboard, and some other needed hardware. At the time of purchase the Pencilbox was a very nice piece of hardware that was relatively convenient, easy to use, and inexpensive.

Before 1989 students were assigned laboratories to do on the Pencilbox hardware that started with relatively small circuits and progressed to much more extensive circuits. These more complicated circuits required the students to design and build such things as data processing and control circuitry for a serial adder.

When the Digital Logic class was taught in the Spring of 1989, the students began using the CircuitMaker software to perform their laboratories. This software made the laboratory less frustrating for the students and helped improve the teaching of the Digital Logic course.

Advantages and Disadvantages of the Pencilbox Logic Designer

The use of the Pencilbox hardware is very valuable for students because it gives them valuable hands-on experience. However, it has a very big disadvantage when trying to wire more complicated circuits. The pins on the IC's are very close together and the chances of forgetting a wire connection or placing a wire in an incorrect spot becomes very likely, especially when the circuits become more complicated. Also the chances for intermittent connections or faulty IC's complicates the problem still further. The students often experienced a great deal of frustration using the Pencilbox hardware on more complicated circuits and this handicapped their learning of the material.

The CircuitMaker Software

CircuitMaker is a software package written and sold by MicroCode Engineering. This company is located at 1943 North 205 West, Suite 1, Orem, Utah 84057.

CircuitMaker runs on a Macintosh computer with a mouse and at least 512K of main memory. A hard disk drive with at least 350 K of free storage is also nice, but it can be run from a floppy disk. The CircuitMaker software allows the design, drawing, and simulation of digital electronic circuits. An instruction manual for CircuitMaker and the software with all the basic functions costs $100. It is also possible to buy enhancement packages that include a TTL library for $60 and an extension that allows the creation of macros for $80. An educational institution can purchase all three packages for $200 or a site license for one room of Macintoshs costs $550.

The Macintosh mouse is used in drawing up the circuits and the CircuitMaker software has the usual icon display with point and click menu selection. Details regarding the use of the CircuitMaker software are discussed later on in this paper.

CircuitMaker Advantages and Disadvantages

The are several advantages in using this software program over using the Pencilbox hardware.

- The students have much greater success using the CircuitMaker. Some of the reasons for this are:
    - the connections are easier to see and the wires are easier to lay out in a more straightforward and logical way. Therefore the chances are much better for the students to be accurate with connections and the chances of forgetting a connection are reduced.
    - a greatly reduced chance for bad connections. It is possible to have faulty connections with the CircuitMaker software, however, the software provides a debugging function that draws wires as solid lines when they are logically true and as dotted lines when they are logically false. A wire that is not connected will usually flash between these two states. Once this is understood by a user, there should be no undetected bad connections.
    - all IC's and electronic components should function correctly. Assuming that all components in CircuitMaker were programmed correctly to start with, they should continue to work properly no matter how many times they are used.
- Much larger circuits can be built with the CircuitMaker software than with the Pencilbox hardware. The overall space for circuitry is large and if the creation of macros is included, very large circuits can be built. Macro creation will be explained later on in this paper.
- It is possible to save all created circuits for use at a later time. Using the Pencilbox hardware, completed labs are torn down to make room for new circuits. The ability to save old circuits along with the ability to create a macro device from a previously created circuit makes for a very powerful tool.

There are a couple of disadvantages in using this software program.

- The students may feel somewhat isolated from the hardware, since they never actually are able to touch any hardware using the CircuitMaker.
- There are at least a couple of bugs in the software, which will be described later. Presumably these will be fixed in the future, however, these can be very frustrating for students if they aren't aware of them.

CircuitMaker in More Detail

To run the CircuitMaker software, click on the CircuitMaker icon from the Macintosh operating system environment. The program will be loaded and the CircuitMaker menu will be displayed. There are several menu selections displayed across the top of the screen. The menus are: File, Edit, Macros, Options, Lib1, Lib2, Lib3 and perhaps Lib4, Lib5, and Lib6 if the extra TTL library software has been purchased.

File Menu

The File menu allows the user to save a circuit, load a previously created circuit, quit the CircuitMaker program, print a circuit, and other related operations.

Edit Menu

The Edit menu allows the user to make changes in a circuit. It contains operations to Move a circuit component, Label a circuit component, Rotate a gate, Wire, and other required editing in a circuit. The Wire option allows components to be wired together. MicroCode Engineering, the developer of CircuitMaker, is in the final testing stages of an option that will allow components to be moved and keep the component wires still connected.

Macros Menu

A macro is a way of creating additional library devices by having the user encapsulate a circuit into a "black box". This "black box" is called a macro device and can be selected and included in a circuit as any other electronic device can be included from one of the Lib menus. The Macro menu allows the user to Define a new macro, Save a macro, Expand a macro, or Delete a saved macro.

Options Menu

The Options menu contains choices such as: Trace On/Off, Single/Double Click=Turn, Waves On/Off, Repeat On/Off, and Run/Pause. All of these options are toggles. A toggle is like a switch that when pressed once selects some option and if pressed again, reverts to the original state. So for instance if the Trace is Off, then highlighting this option and releasing the mouse button will turn the Trace On. Some of the menu choices under the Options menu deserve further comment.

The Trace On is the debugging option that was mentioned earlier. When the Trace is On, each wire in the circuit that is logically true will consist of a solid line, whereas those that are false will be dotted. A wire with a bad connection is indicated when the wire flashes between a solid and dotted line.

When wiring components together, and the Single/Double click=turn, is set to Single, a single click of the mouse button, causes the wire direction to be perpendicular to what it previously was. When in this mode, a double click exits the wiring option. If, on the other hand, this option is set to Double, a double click causes a change in wire direction and a single click of the mouse button causes an exit from the wiring option.

In the Lib1 menu a user can select an oscilloscope to connect to a wire. If the Waves On/Off option is set to On, the bottom part of the monitor screen will display the square waves generated when logical signals travel through the monitored wire.

The Repeat On/Off option when set to On allows an electronic component to be selected from one of the Lib menus and be placed multiple times by clicking the mouse button when the component is at the desired position, without going back to the menu. A double click of the mouse button will exit from the particular library component selection. When the option is set to Off, a single selected component from a library menu can be placed by clicking at the required position. To place another component in this configuration, another selection must be made from a library menu.

Under the Run/Pause option, the Run will cause the execution of the circuit simulation. When the Pause is selected, the simulation is stopped.

LIB1, LIB2, AND LIB3 Menus

These menus have the electronic components that come with the basic CircuitMaker package. Some of the components that can be found under this selection are: Vcc (5 volts), Ground, Switch, LED, Oscilloscope, Seven Segment Display with Decoder, AND gate, NAND gate, OR gate, NOR gate, XOR gate, and various Medium Scale Integration (MSI) TTL circuits such as a parallel adder, a multiplexer, and a flip-flop. There are a total of thirty-five different circuit devices contained under Lib1 and Lib2. Lib3 contains an additional nine circuit devices.

LIB3, LIB4, LIB5, and LIB6 Menus

These menus contain additional TTL integrated circuits. Under the basic CircuitMaker package Lib3 has only nine circuit devices but with the enhanced version it contains an additional eight more devices. There are a total of 71 different circuit components contained under the Lib3, Lib4, Lib5, and Lib6 menus. Besides a large selection of MSI gates, there is included a 1 Kbyte RAM chip and a 32 byte PROM chip. The PROM can be programmed under the Edit menu.

Integration of CircuitMaker into the Digital Logic Course

In the Digital Logic course students are required to perform the easy laboratory circuits at the beginning of the course using both the Pencilbox hardware and the CircuitMaker software. This avoids the problem of students not really having hands on experience with the hardware. It also gives them a chance to become familiar with the CircuitMaker software on very easy circuits. As the circuits become more complicated, the students are no longer required to do the circuit with the Pencilbox hardware. By requiring the students to use both methods at the start of the laboratory and then allowing them to use only the CircuitMaker the advantages of both methods are exploited.

Bugs and Problems in CircuitMaker

At least two observed problems have been detected in using the software. In some instances it is not possible to retrieve a circuit that has been created and saved to disk. Apparently a circuit that is too large for the remaining disk space is not saved properly, so that it is not possible to retrieve it at a later time. The only solution at this time is for users to observe how much space was required for circuits that have been built in the past. Then when a new circuit is to be built, the user should make certain that the disk contains twice as much free space as was used for a previously saved circuit.

The other observed problem was with the J-K flip-flop, 74LS109. If the set-reset switches are connected to Vcc (5 volts) as normally required for the flip-flop to work properly, the flip-flop will not start and will not change states. To avoid this problem, at least one pin, either the set or the reset should be connected to a switch and flipping the switch off and then on will allow the IC to begin changing states.

When discussing these two bugs with MicroCode Engineering, the company expressed a real concern and indicated efforts would be made in correcting them. Also in the discussion at that time it was concluded that there are probably other errors in the software.

Though it is not a bug, it was found that having the hexadecimal decoder attached to the seven segment display was inconvenient. It sometimes would be helpful for the students to create the decoder on their own or design another decoder that might generate symbols other than the hexadecimal characters. A way around this is to have the students wire several single LED's in a straight line segment, and then several of these segments can be used to create shapes other than the hexadecimal characters.

Student Perceptions

The students have really enjoyed working with the CircuitMaker software, especially as contrasted with their work on the Pencilbox hardware. It is fun to construct the circuits and see the results of what they have learned in the lecture portion of the class. Because of the ease in wiring circuits, the students are willing to be creative and try out ideas that would not have been possible using the hardware approach.

The Future

Because the goal of the Digital Logic course has been the understanding of a computer, the building of an elementary CPU would be a valuable exercise. CircuitMaker now makes this possible. During the Spring 1991 offering of the course, the students will build a very elementary CPU. The intention is to make it a four instruction machine with an Instruction Register, Program Counter, an Accumulator Register, and an external memory with a few instruction program. Once students have finished with this project, they should have a very good understanding about the operation of a computer.

Summary

In summary, the CircuitMaker software has many advantages in the design, drawing, and simulation of digital circuits. Among the advantages are: ease of use, relatively large circuits can be accommodated in part because of the ability to do macros, and the ability to save a created circuit for use at a later time. All of these pluses make it an ideal program for the teaching of Digital Logic, but also it could be used in other electronic type courses or even in the design of digital circuits for manufacture.

Bibliography

[Man 1984] M. Morris Mano, *Digital Design*, Englewood Cliffs, NJ: Prentice-Hall, 1984.

[Rot 1985] Charles H. Roth, JR., *Fundamentals of Logic Design*, 3rd ed. St. Paul, Mn: West, 1985.

[Boe 1988] Ozzie Boeshans, *CircuitMaker User Manual*, Orem, UT: MicroCode Engineering, 1988.

Knowledge Gateway Project

Eric Jensen
Wenatchee High School
Wenatchee, Washington

Abstract

The purpose of the project is to study the effects of concentrated computer use on the language acquisition and intellectual confidence of at-risk students.

Targeted students include those having limited reading skills, who are "at risk" in terms of completion of the high school curriculum, who do not use English as their first language, who lack confidence in their ability to self-direct their learning.

Participating teachers schedule classes into the networked computer lab for successive days of intensive use focusing on reading and writing skills. Lessons include progressive skill building in the manipulation of text and graphics.

As a result of this project, student writing is changing from a focus on personal thoughts to include teacher suggestions and information from other resources. Graphics are assuming an important role in motivating writing and developing ideas. Graphics help students visualize thoughts that can be difficult to express in words. The graphic interface of *HyperCard* appears to make some ideas more accessible to students. Learning activity at the computer is often focused for extended periods of time. Students familiar with *HyperCard* fundamentals at the "browsing" and "typing" levels find the transition to different HyperCard based programs easier. Students are practicing information skills by using *HyperCard* in conjunction with the AppleShare Network. Students are beginning to initiate action on assignments from other classes that allow them to utilize the computer as a "power tool." The technical quality of student writing improves as time on the computer and familiarity with "core" software tools increases.

LEGO TC Logo—Fourth Grade Science Becomes Real

JoAnn F. Karaffa
Spring Mill Elementary School
Indianapolis, Indiana

Abstract

In 1988 I was a recipient of a small state technology grant, which enabled me to develop a program that utilized LEGO TC Logo within my science curriculum. The project objectives included building models of machines to test scientific concepts of work and the effects of energy on moving objects. Investigations of simple, compound, and programmable machines demonstrated these concepts. A strong emphasis to incorporate the methodology of Scientific Method was used. Students were involved in a variety of learning and working group strategies such as cooperative learning, team cohesiveness, simulated work, forces, and problem solving groups.

Once students began working with programmable machines, they found a need to use logical reasoning and sequencing skills to successfully program models to function using Logo. The children became better communicators of ideas, and found that discussing problems and asking others to help brought them quicker success. When it was time for the school Science Fair and Invention Convention, the students quickly came up with ideas.

Since the original grant, curriculum plans have been expanded and utilized to continue to enhance the learning environment of my classroom. Students are eager, enthusiastic, and look forward to LEGO days.

Curriculum Oriented Computer Resource Guide for Middle School

Mary Grace Jaeger and Cheryl L. Hepp
Jefferson County Public Schools
Louisville, Kentucky

Abstract

Middle school teachers in the Jefferson County Public Schools (Louisville, KY), aided by staff from the Computer Education Support Unit, developed a computer resource guide that is based on the curriculum taught in grades six through eight. Lessons in the four major content areas—language arts, math, science, social studies—are included for each grade level. Since emphasis is placed on writing in all areas of the curriculum, writing activities are included in every lesson. Lessons also incorporate the use of databases and spreadsheets.

All materials needed for a particular lesson are included in the resource guide. A disk that contains files for lessons, as well as printed copies of all files on the disk, is also included.

This resource guide is an outgrowth of the New Kid in Middle School project in which a networked 32-computer lab was placed in each of the 24 middle schools in Jefferson County. Ten teachers from each middle school received 60 hours of training that emphasized the use of the computer as a tool for integrating writing across the curriculum.

A short video-taped lesson is available to teachers to introduce the guides and to explain their use. This video-tape is part of the MICROTECH tape series compiled by the Computer Education Support Unit. The tape series consists of instructional tapes that are less than 15 minutes in length.

Hypercard: Reinventing the School

Joseph Hofmeister and Joyce Rudowski
Cincinnati Country Day School

Abstract

As the winds of educational reform blow through the country, one thing is very clear: We are not satisfied with the way things are going. The path to reform at our school has had three identifiable steps, so far.

1. As technology—specifically the Macintosh, *HyperCard*, and multimedia—have begun to infuse the school, we discovered that student involvement with the hardware added an unusual amount of excitement to a course. Forging a path of excitement through the school using technology in this way, focusing on students rather than teachers, became our goal.
2. As the momentum of the process grew we realized that the critical issue was the fact that students were creating their own knowledge rather than having it fed to them piecemeal by their teachers. The relationship of the people in a classroom to a certain body of information was changing. The symbol of the hourglass seemed appropriate to what we had had, with information passing through the narrow passage—the teacher—to the students waiting for the information in the bottom of the hourglass. A more appropriate symbol for our new model of the classroom—a student-centered one—is the sieve, with both teachers and students underneath, poking new paths for the information.
3. The unanticipated new level of development that occurred in the second year of the project was the movement toward the authentic task. Assignments and projects in the student-centered classroom seem to move inexorably toward the real. Students who are engaged in their studies want to do things that actually are important. Textbook problems whose importance extends only as far as the boundaries of the course itself become less interesting and motivating.

This session details a number of examples and projects that have come out of these classrooms and highlights some of the discoveries and problems with which the presenters are dealing.

Successes Despite Extremely Limited Computing Time in a Primary Classroom

Sandee Hedetniemi
Department of Computer Science
Clemson University
Clemson, South Carolina 29634
(803) 654-9650

Helen Robinson
Morrison Elementary School
Clemson, South Carolina 29631
(803) 654-2341

Abstract

This paper addresses the constructive use of very limited computing time in a primary classroom. A second grade class of 22 students was restricted to having the exclusive use of an Apple IIGS computer for a period of nine weeks during the school year. This nine weeks was broken into a six week period in the fall and a three week period in the spring. This limited time frame was dictated by the fact that there were four second grade classes which shared the one computer.

Curriculum material was developed which would allow the actual time that the students used the computer hardware to be an extension of activities with which they were familiar. Emphasis was placed on reinforcing skills that are normal expectations of second graders; introducing the problem solving technique of determining subgoals; and becoming familiar with computers.

In an article in USA Today newspaper it was reported that of 1100 teachers surveyed, 59% say that teachers are inadequately trained in computer use yet 82% of the teachers say that computers increase student motivation [7]. This paper is directed towards those teachers who feel that their computing equipment is so limited that they cannot constructively use a computer in their class. The material in this paper is a result of such a situation. A second grade class of 22 students was to have quite limited access to computing facilities. This particular class was to be allowed the exclusive use of an Apple IIGS computer for a period of nine weeks during the school year. This nine weeks was broken into a six week period in the fall and a three week period in the spring. This limited time frame was dictated by the fact that there were four second grade classes which shared the one computer.

At no time was it the intention of this project that a scientific study be made on the effective use of limited computer facilities. This would have prompted the need for a control group who would be totally denied use of the computer. This would make a bad situation even worse for those students.

It should be noted that most of the students had little previous experience using a computer. Approximately a quarter of the class had computers in their homes, largely as a result of their parents needing the computer for their own work. Four of these students, having demonstrated their familiarity with computer interactions, served as "teachers" for the remaining students. Of the remaining 16 students, approximately a quarter (4) had used the computer in the first grade over a period of several weeks to play educational games. The remaining 12 students had at best brief encounters with a computer. Most were quite content to think of a computer in the same terms as they thought of TI's Speak and ... Series. That is a computer is a device which is controlled by buttons pressed by the user; that can be expanded by the insertion of additional cartridges; and that has a limited repertoire of capabilities. However, all the students brought with them great excitement over the chance to use a computer.

The primary goal of our efforts was to constructively take advantage of the computer for the very limited time that it was available to the students . A secondary goal was to enhance the student's familiarity with computers. This created an additional secondary goal—to encourage the students to realize that "computing" is more than sitting in front of a computer. We felt that we could best accomplish these goals by using the computer to extend the skills that the students needed in their daily subject matter. The subject matter fell in the following general categories: problem solving, mathematics, social studies, language arts, and science.

In agreement with the National Council of Teachers of Mathematics [6] focus on problem solving and the National Council of Supervisors of Mathematics [5] belief that "learning to solve problems is the principal reason for studying mathematics", we feel that problem solving is fundamental to learning and living. Yet it is both difficult to teach and difficult for students to learn. However, students have a much better chance of solving problems if they have a "grab-bag" of techniques to apply to a given problem. Young children are often overwhelmed by a problem statement when in fact they have the skills to solve the problem. We chose to emphasize the problem solving technique of subgoals, i.e., breaking the problem into smaller problems which can be solved independently, cf. Polya [10]. Our goal was to present this technique to the students in classroom presentations and then emphasize this technique during their interactions with the computer. Although we did not have sufficient time with the computer to allow the students complete freedom to explore with Logo, as Papert [8] suggests, we did use Logo in a structured environment.

Over the past fifteen years, thoughts on the appropriate use of computers by students in the classroom, cf. Billings and Moursund [1], have taken on several forms ranging from programming by all students, cf. Luehrman [4] to using preprogrammed software, cf. Bork [2]. The use of preprogrammed software might involve students using the computer for drill and practice, students using programs which help them learn, computer assisted learning, and students using the computer as a tool, eg. with word processors or spreadsheets. Our emphasis was primarily to use the computer to help students learn but not in the sense of computer managed instruction. The computer was used in a fashion similar to other manipulatives which can be found in the classroom, such as place-value counting blocks or geoboards However we also allowed the students to "program" at a very high level, i.e., subprogram calls in Logo and used two software packages which provided practice.

With the primary goal in mind, a blackline master of the computer keyboard was constructed using MacDraw on a Macintosh II. Over a course of six weeks the students were given many opportunities to become familiar with the keyboard by coloring keys with symbols from their name, address, telephone number, the emergency number 911, and spelling words. In addition words were taken from the South Carolina Basic Skills Science and Social Studies list. Hence, these items were chosen to reinforce material that they should already know or were currently learning.

It is our belief that second grade student's hands are not large enough to handle touch typing. However, it is clear that familiarity with the keyboard is fundamental to appropriate use of limited computer facilities. Furthermore, if computing facilities are limited, economics make it hard to justify using the computer to learn the keyboard. A more economical method would be to purchase several of the electronic devices which help teach touch typing such as Type-Right or Precomputer 1000. However, Carney [3] suggests that Type-Right might not be a suitable alternative without developing supplemental material to that which comes with its purchase. Similarly, the use of typewriters during creative writing exercises would also be useful. However, if finances do not even allow these luxuries, the use of blackline masters is better than no familiarity.

We engaged in several class presentations before the student's window of hands-on computer use which helped them to become familiar with computer components and computer terminology. It was hoped that the students would feel comfortable using the computers as a tool.

The first such presentation was a hands-on opportunity with the physical components of a computer system. Each of the students was given a packet of precut pictures of various components such as the monitor, disk drives, keyboard, printer, circuit board, and the computer system as a whole. [Note: an ample number of these pictures can be found in magazine advertisements or flyers from the computer companies. Apple has a particularly nice flyer with the individual computer components identified.] These pictures were used to create their own picture dictionary. We had some of these same components available for them to look at and touch including a floppy cut open and a mouse. Even though a quarter of the class had a "computer" at home, most of the students did not appreciate the term computer system, i.e., computer, monitor, keyboard, etc., taken as a whole. The real attention-getter was an Imagewriter, which through its self-test, could print out pages while only plugged into a wall.

The second presentation addressed the software behind the use of the computer. In this presentation we wanted the students to become aware of the concept that a computer has no intelligence of its own. Fundamental to the correct operation of the electronics is the concept of an algorithm—an orderly, well prescribed method for action. We did this through the analogy that recipes have three major components—input (ingredients), algorithm (course of action), and output (finished product). We made the point that a change in either of the first two components, i.e., ingredients or course of action created a different finished product. We also emphasized that the computer could not make judgment calls as some recipes require humans to do. [For example, box cake recipes tell us the cake is finished with the cake springs back when touched lightly.] The class participated by making "play dough" from a recipe. Three things were accomplished by this exercise. First, as they tried to add just the right amount of water they began to appreciate the necessity of preciseness. Second, the students' skills of following precise written and oral directions were reinforced. Third, because of our discussion of how "inadequately" many recipes are written, they began to realize that their inability to do something was not necessary a result of their capabilities but rather a result of the request not being clearly stated.

The third and fourth presentations were directed toward algorithms for robots—human robots. These presentations served as nice follow-ons to the discussion of algorithms and as precursors to the language Logo. As we mentioned before, it was never a goal to teach programming, but rather to teach about programming. A fourfold goal was accomplished—the students began to appreciate the concepts which cause the computer to execute algorithms; the students would be more adept at orienteering; the students had opportunities to follow instructions and the students developed their sequencing skills.

The third presentation concentrated on relative directions, i.e., given the direction that the robot is facing should a left or right turn be made. Turns were expressed as 1/4 turns of a circle. Many of the students were familiar with full turns, half turns, and quarter turns through gymnastics and diving (either personally or having viewed the Olympics). The courses which the human robots were to follow were drawn as connected blocks on a shower curtain liner. Each block corresponded to the movement of one step of the robot. The only necessary commands were Forward, Left, and Right. The students took turns serving either as robots, footprint makers, algorithm directors or observers. The goal of the algorithm director was to direct the robot to walk across a given path taking the correct number of steps between turns and making the turns in the correct direction. The robots were instructed to follow instructions of the algorithm director exactly and were not to make corrections of their own. The footprint makers left laminated footprints [constructed using MacDraw on a Macintosh II and resembling "Bigfoot"] so that the actual path the robot took under the direction of the algorithm director could be seen by the other students serving as observers. Many of the students found it difficult to decide the correct turning direction without placing themselves so that they were facing the same direction as the robot. This is of course the same problem one has in working with maps.

The fourth presentation was similar to the third except that letters were added to the blocks. The robots were given an additional command—report. During the student participation, the algorithm director directed the robots to follow the path correctly and at the same time make appropriate stops for reporting the letters to spell some of their spelling words. By providing "algorithms" to the groups, the students were given the chance to practice both deciphering and reading their vocabulary words. The students at this time also learned the correct algorithm for the robot to leave a trail corresponding to a rectangle and a trail corresponding to a square. These algorithms would be followed up in the actual use of the computer.

It should be noted that the activities associated with the third and fourth presentation can be performed many times and that they are excellent activities for rainy days when an outdoor recess is impossible. If the letters on the path are simply taped on, then the letters can be adjusted to spell whatever the current word lists necessitate.

At this point the students reached their six-week use of the computer. Since this was during Computer Learning Month we took the opportunity to enter The Learning Company's silly story contest using Writer Rabbit. We began by introducing the terminology of the "who, what, when, and where" composition skills in descriptions of their regular class work. Then to best utilize limited computer time we used a creative writing session (and templates which appear in the program) to allow the students to think before they entered their story in the computer. The opportunity to enter their story in the computer was a positive promotion of the enjoyment of creative writing. After all the students had a chance to enter and print their stories, they "played" some of the learning games which available in the program. These games reinforced their composition skills. The students were also given opportunities to create and print different types of stories—narrative, descriptive, and explanatory.

To facilitate efficient use of the computer software, we used a core of four students to teach the other students the mechanics of the software use. These students were above grade level readers and generally finished their work early. Not only did this teaching strategy extend the usable time for the rest of the students, but it also gave these four students valuable opportunities to improve their communication skills. If time had been less critical, all students would have been given the option of becoming "teachers".

The fifth presentation introduced the problem solving technique of creating subgoals to achieve a complicated goal. In terms of Polya's model of problem solving [9], we wanted to devise a plan where the subgoals were to be performed in a sequential order, similar to a sequence of additions and subtractions in many math problems. The vehicle was the creation of the tangram pieces in their standard presentation format as a square. The students first needed to identify the component shapes of the tangram and decide just what was an appropriate order to construct the pieces so that they filled the square. Not only are students allowed the opportunity to identify and create the more standard geometric shapes of squares and triangles, but they also had the added bonus of being introduced to the parallelogram shape. The first subgoal was to create the exterior square which served as a boundary in which the remaining pieces were to be drawn. Achieving this goal relies directly on their previous knowledge as human robots. Actual creation of the remainder of the pieces requires algorithmic thinking that had not previously introduced. However, this was not the goal of the activity.

Since the class had the computer for their use, the emphasis was on seeing the subgoals executed in a sequential manner. Each student used a Logo program which had been written with subprograms corresponding to the creation of the various pieces. Hence, the students "programmed" the turtle by using these subprogram commands. They were shown details of the Logo subprogram to create the exterior square so that they could see how closely it resembled the algorithm they used for their robot's square. [The algorithms were identical except that the 1/4 turns had been replaced with 90° and the turtle Forward command had been substituted by a subprogram SLOWWALK which caused the turtle to move more slowly.] They simply typed the subprogram names and watched the turtle execute each of their "high-level" commands. Here the emphasis was on cause and effect, i.e., that algorithms of commands directly affect the workings of the computer.

However, the emphasis could be switched to a more mathematical content. For example, the square is basically divided in half to form two triangles. One of the triangles is split in half to form two smaller triangles. Each of these topics can be discussed first with paper manipulatives. Then the problem solving technique can be altered from a strictly sequential process to a sequential process where some of the steps contain a hierarchy. In this case subgoals refine further subgoals. While this discussion may be beyond second graders, it certainly can be introduced in later grades.

The sixth and seventh presentations were a combination of two activities, previous use of the robot on the shower curtain and the opportunity to practice unit measurement. The path that a robot (if the computer is not available) or turtle can also be measured in terms of the footstep for the robot and a corresponding "pace" for the turtle. A "pace" of the turtle correspond to the movement from one edge of a block to the opposite edge of the block. Hence, blocked paths similar to those on the shower curtains were created for the turtle to move through on the screen. A footstep of the robot or the "pace" of a turtle corresponded to a unit of measure which were taken to be one foot and one centimeter, respectively. It should be noted that all the students were happy to pretend that those were legitimate units even though the turtle's one centimeter was easily 2 1/2 centimeters.

Initially, mazes were created for the turtle to walk through by coloring large portions of the screen (using the Fill command) leaving very simple paths between the walls. [In reality, maze is a misnomer since there are no wrong turns.] The paths were divided into blocks to facilitate the measurements along the turtle's path. It should be noted that but future work would require that the turtle follow an outside edge of a block. [This was introduced by having the blocks show white while the line the turtle was to follow show blue. As the turtle covers the path its color changes to orange.] The first path was straight to emphasize the one-to-one correspondence to the number of "paces" the turtle took and the number of centimeters. The remaining paths became successively more difficult. The emphasis was not on programming the turtle moves but rather on interpreting the path the turtle took to reinforce their mathematics work—addition and subtraction skills. Questions were phrased in terms of north, south, east, and west directions to reinforce orienteering concepts. Questions generally dealt with simple word problems such as which direction did the turtle walk farther or how much farther did the turtle walk east than west. Here it is particularly useful to discuss how a turn to the left and a turn to the right may both result in the turtle pointing northward.

As the mazes became more complex, the need for subgoals became more apparent to the students. Again the subgoals were to be executed in a sequential manner. An appropriate subgoal consisted of reaching the next turn in the path. A subgoal was accomplished by a turn followed by the correct distance to reach the next turn. [Similarly, the turn can be the last step of the previous subgoal.] In fact, the maze was created using subgoals; when the maze is first drawn on the screen it is drawn in sections. Furthermore, these subgoals could be expressed in terms of up/down, left/right, and above/below relationships.

It should be pointed out that while the emphasis was on measurement, to reinforce the mathematics unit of study, the students did complete part of the instructions for the turtle. This reinforced the instructions for the robot with which that they had already become familiar. The students were given handouts containing copies of the mazes where scattered commands were missing. This also had the effect of emphasizing the direct relationship between a command of moving a certain distance and the turtle's act of moving that distance.

The seventh presentation was an extension of the turtle's movement through mazes to the turtle's movement around the perimeter of a shape, in particular a square or a rectangle. As before the turtle was constrained to "paces" which corresponded to the movement across a block. In this case the starting screen was blocked into small squares. The students were already familiar with the correct commands because of the robot and of the tangram. Although the emphasis was on measurement skills, we did reinforce the creation of the geometric shapes—squares and rectangles. Discussions first centered around measurement of the perimeter of the shape. Then the students moved on to a discussion of the area enclosed by the walk of the turtle. Since the screen was blocked into unit squares, measurement of the area simply involved counting the squares. The students had already been introduced to multiplication so the discussions also reinforced this concept. The students were given a chance to create rectangles of a given area, first by being given one of the sides and later by having no constraints on any sides.

Because the measurement unit included manipulation of money we encouraged the students to use Scholastic's Microzine Jr. program, Ani-Mall. This program allows the students to practice choosing coins whose sum is a given number in the context of providing change for customers in a mall.

In summary, we can offer the following guiding thoughts about our efforts to utilize limited computer time. Because a computer was not present on a daily basis for the entire year, we could not just wait for the computer to "fit in" to our daily routine. Instead, we needed to modify our curriculum so that computing and

		Keyboard Familiarit	Computer Hardware	Recipe (Algorithm)	Robots I	Robots II	Writer Rabbit	Tangram	Measurement I	Measurement II	Microzine's Animal
Problem Solving								●	●	●	●
Mathematics	Addition/Subt./Mult								●	●	●
	Simple Fractions				●						
	Area / Perimeter									●	
	Word problems								●	●	
	Linear measurement								●		
	Counting Money										●
	Geometric shapes					●		●		●	
Language Arts	Creative writing							●			
	Spelling	●				●					
	Parts of speech						●				
	Computing terminology	●	●	●							
Social Studies	Map skills				●				●		
	Vocabulary	●									
Communication skills	Following directions										
	Oral		●	●	●						
	Written	●					●				●
	Giving directions				●	●					

Figure 1. Matrix of Skills Strengthened

computing concepts became a part of it, if only temporarily. We chose to concentrate on one specific problem solving technique, subgoals. The students saw this technique applied to problem solutions in the classroom and on the computer. Hopefully, this multipresentation will encourage the students to believe that it is a technique suitable for widespread use. As the matrix given in Figure 1 suggests, we used the presentations and computer software to strengthen skills which are normally appropriate for this grade level.

There are two reasons that our discussion may seem strongly positive. First, when one begins with very little, any gain seems like a positive step. Second, in retrospect we would not eliminate any of the topics that we discussed. We did note that many of the children needed for the "mazes" to become more complex at a much slower pace than they did. Therefore, we would provide the opportunity for the majority of the students to have a much more gentle introduction to "navigating" the mazes.

We believe that our work is just a beginning. Although it is unlikely in the near future that the amount of computing time will increase significantly, we at least can extend the activities away from the computer. We would like to introduce two other problem solving techniques—working backwards and case analysis. We feel that the use of "mazes", be they on a shower curtain or on the computer, can be an asset in emphasizing working backwards. By using problems such as 3 plus what number equals 8 on the mazes, we can get the student to select the correct entrance by working backwards from the exit. We feel that the work on area and perimeter used in our second measurement activity can be extended to include the problem solving technique of case analysis. We can use the Logo turtle or our human robots to determine all the ways to create a rectangle of a given area or perimeter.

References

[1] Billings, K. and Moursund, D., Computers in education: A historical perspective, *SIGCUE Outlook, 20,1* (1988), 13-24.

[2] Bork, A. Personal computers for education, New York. Harper and Row. 1985.

[3] Carney, C., *The Computing Teacher, 15*,7 (1988), 48-50.

[4] Luehrmann, A. Computer Literacy—What should it be?, *Mathematics Teacher, 74* (1981), 682-686.

[5] National Council of Supervisors of Mathematics, Position paper on basic mathematical skills, National Institute of Education, Washington, DC, 1977.

[6] National Council of Teachers of Mathematics, An agenda for action, NCTM, Reston, Va., 1980.

[7] Ordovensky, P., Teachers say they don't compute, *USA Today*, August 28, 1989.

[8] Papert, S. Mindstorms: Children, computers and powerful ideas. New York. Basic Books. 1980

[9] Polya, G. How to solve it. Princeton. Princeton University Press. 1957.

[10] Polya, G. Mathematical discovery: On understanding, learning and teaching problem solving (Vol. 1). New York. Wiley. 1962

The Willingness of Rural Iowa Educators to Participate In Computer Teleconferencing

Lowell Monke, MA
Sharon E. Smaldino, PhD

Curriculum & Instruction
University of Northern Iowa
Cedar Falls, IA 50614-0606
(319) 273-3250

Abstract

This study investigates the willingness of rural educators in Iowa to use computer teleconferencing. Over two hundred educators from 36 schools were surveyed. All superintendents, all principals, and four teachers from each of the participating schools were included in the study. The survey consisted of a two part questionnaire, one part dealing with professional information and computer use, the other with attitude toward using computers and participating in teleconferencing.

The results indicate a very positive attitude toward computers in general, a high level of computer availability, and a broad interest in using computer teleconferencing, but limited knowledge or experience in actually working with computer teleconferencing. The overall assessment was that there appears to be a high level of willingness among all three categories of educators to use computer teleconferencing. There was also an indication among these educators of a willingness to participate in pilot projects involving computer teleconferencing.

Iowa is a representative mid-western state with both urban and rural school districts. The results of this study would be of interest and value to educators in other states.

Introduction

Rural Iowa schools face a growing financial struggle to maintain their independent identities. Many of the problems facing these small schools are perennial and unique. Teacher isolation, lack of resources for staff and students, inability to provide programs for special needs students and general inefficiency are a few of the problems listed by Nachtigal (1982). While consolidation is the most commonly proposed solution to these problems, there are many, especially the residents of the rural communities, who defend the benefits of rural education and see reasons to fight the trend toward large centralized education. Many rural educators and parents seek ways to overcome the educational disadvantage of small numbers of students without sending them out of the community.

One avenue being explored is the use of technology to bring additional resources to the schools. Although interactive televideo is the most widely publicized of these innovations, other smaller scale, less costly communication systems are being investigated as well (Schrum, 1988).

One of these avenues is computer teleconferencing. Computer teleconferencing has been defined in this study as activities involving computers in which users interact with each other and with the information they are providing each other. It does not include activities such as using a mainframe accounting program or using centralized student management systems.

Computer teleconferencing has been used for some time by universities and large metropolitan schools in Iowa and across the nation to move information quickly from one building or department to another. The hardware requirements are minimal, a microcomputer and a modem. Computer teleconferencing, therefore, is relatively inexpensive and easy to start up. With the exception of Alaska, where it is an integral part of providing educational resources to remote areas, most of the computer teleconferencing to date has been reported in large, urban schools (Sequin, 1988).

The use of computer teleconferencing specifically to aid rural schools has only just begun to receive attention. The potential benefits of computer teleconferencing among rural schools have been articulated (Azarmsa, 1987; Birkhead, 1986). They include more efficient administrative communication, greater teacher access to educational resources, and increased interscholastic interaction among students. A few programs have been initiated through the Iowa Area Eduction Agencies and universities that include rural schools, but these have a narrow focus and mostly local coverage.

Purpose of Study

It is doubtful that computer teleconferencing will become a valuable tool for rural schools without those who will be in the positions to take advantage of it expressing a willingness to put it to use. Lawton and Gershner (1982) suggested that the success of any computer-based project is dependent upon participants' attitudes toward computers. Without teachers willing to accommodate change by adapting instruction, any procedural or curricular change has little chance of success (Moursund, 1979). Staff acceptance, or lack of it, will play a crucial role in determining whether computer teleconferencing becomes a valuable resource for rural schools.

The survey of rural Iowa educators conducted in this study investigated the current interest in computer teleconferencing among rural Iowa schools and provides information on the extent to which these educators are willing to use this developing technology.

Method

Educators from 36 schools participated in the study. All schools were located in the state of Iowa and met the following criteria:

1. A total enrollment, K-12, of under 1000 students
2. Located at least 10 miles from any metropolitan area with a population of greater than 50,000

Three categories of participants were identified, teachers, principals, and superintendents. All principals and superintendents in the participating schools were included. Four teachers from each school were asked by their superintendent to participate, two from the elementary level, two from the secondary level.

Two hundred thirty-seven surveys were mailed. By category 36 superintendents, 57 principals, and 144 teachers received questionnaires. One hundred seventy-seven surveys were returned, 33 superintendents, 42 principals, and 102 teachers.

A questionnaire was developed specifically for this study. Two versions were developed, one for administrators and one for the teachers. Each version was developed around four basic components:

1. the computer environment in the work environment
2. general attitude toward computers and interest in using them in the educational environment
3. needs perceived by the participant that could be addressed by teleconferencing
4. interest expressed directly related to computer teleconferencing

Both versions contained parallel components. Any variations occurred primarily in addressing the terms to the proper category of participants. One component was addressed to administrators only since it approached the issue of willingness from a different angle. This component focused on the perception that the administrators had of the computer attitudes of the teachers as a group within their schools. It was not included in the other components of willingness because it referred to willingness of others, not the participant. This component was not included in the teachers' version.

The survey consisted of two main parts. Part One contained questions seeking factual information about computer use, availability, and training. It also contained questions about the personal and professional background of the participants. Part Two was structured as statements to which participants responded by circling the most appropriate number corresponding to a five choice Likert-type scale. The scale ranged from "strongly agree" to "strongly disagree." Three sources were used to model these statements (Lillard, 1985; Simonson, Maurer, Montag-Torardi and Whitacker, 1987; Waggoner, 1987). Three panels of experts in the fields of administration, teaching, and computer teleconferencing judged the survey for content and form. Revisions were made according to their recommendations.

Seventy-five schools were contacted for willingness to participation in the survey. These schools were selected randomly from a listing of all Iowa schools that conformed to the definition of rural schools. Each school superintendent was contacted by mail and asked to participate in the survey. Thirty-six superintendents responded. Survey materials were sent to these superintendents, with the appropriate number of surveys for principals and teachers. There were instructions for distribution of the surveys in the packet.

The superintendents were to select two teachers from the elementary and two from the secondary levels within their district. A suggested procedure was included to assist them in a random selection of the teachers. Assurance of confidentiality was also provided.

Results

The results of this survey will be described in terms of frequency of responses. All of the data received has been included in the analysis. In some cases there were unique situations and so the description of the results will reflect those special conditions.

Professional Characteristics

This study included 33 superintendents, 42 principals, and 102 teachers. Not all participants responded to all of the items. There were some responses that indicated shared responsibilities among superintendent and principal roles in the schools, or multi-level principal positions. There were broad ranges in age and educational background for all of the participants.

There were a vast range of content areas represented. Most of the superintendents came from the social science field, while principals most frequently came from the social science, math, and elementary teaching fields. Language arts, math, business, science, and no special area (primarily elementary teachers) were indicated by the teachers. Two teachers identified themselves as computer science teachers.

Over 65% of the schools surveyed have more than 20 instructional computers being used K-12. Seventy-five percent indicated they had computers specifically designated for administrative purposes.

Only three superintendents identified the presence of computers designated for other activities besides instruction and administration. There was no consistency as to whether there were enough computers in the schools, although elementary principals responded they felt there were enough.

Access to the computers was a problem indicated by nearly 25% of the teachers. About 75% of the teachers indicated they had at least one computer in their classroom. Even if obvious lab situations were ruled out, over half of the teachers indicated they had from one to five computers in their rooms. Only two lab settings were identified. Nearly 60% of the respondents indicated that some computers were on mobile carts. Eighteen of the 33 superintendents indicated that the schools currently owned a modem.

In the area of general attitude toward computers in education, all of the educators' responses were favorable. Nearly everyone who responded indicated that they had some training in computer use. Training varied from self-instruction to workshops and college courses. Teachers more frequently identified college courses as their source of training. Most of the respondents indicated that they felt they did not have enough training in the use of the computer. In the area of teleconferencing specifically, there was a clear indication that the majority did not feel they had enough knowledge about teleconferencing, nor its application in education.

Application of Teleconferencing in Education

While many indicated a limitation in computer training, nearly all of the respondents indicated a lack of knowledge about teleconferencing. Many indicated they were "unsure" of any application in education. There was a definite lack of consistency in responses related to the type of computer teleconferencing activities that might be beneficial to the students.

The educators involved in this survey indicated an interest in trying computer teleconferencing, with 104 out of 175 total indicating they would participate in a pilot project of computer teleconferencing. Still, this indication of willingness to participate has to be viewed tentatively due to their indication of lack of knowledge about teleconferencing and its application in education.

Administrators' Perception of Teacher Attitudes

There is a widespread agreement among all administrators that teachers generally do not have a negative attitude toward computers. There is a split opinion among administrators as to whether teachers are reluctant to incorporate new technology into their teaching methods. Only one administrator felt that there would be no teachers in his school that would be eager to be part of a pilot computer teleconferencing project.

Conclusion

This survey would support several conclusions about rural educators and their willingness to participate in computer teleconferencing. First there is a high level of willingness to participate in computer teleconferencing. This willingness is not limited to the administrators, but includes teachers.

There already exists a number of educators in rural Iowa schools that have had training in computers. They also indicated a positive attitude about computers. Many indicated they did not feel adequately trained, however, nearly all of the teachers indicated they had college courses in computers.

Those responding to the survey indicated a lack of knowledge about computer teleconferencing and its educational applications. Very few indicated they had experience with teleconferencing.

There was no consistency as to the content area or application for computer teleconferencing. Teachers do not seem to have enough of an understanding of the concept and process to be able to make educated decisions as to the appropriate applications of this technology.

The favorable responses from rural educators indicates that the issue of computer teleconferencing be pursued further. It is recommended that prior to any actual project involving the application of teleconferencing in instruction, there first be an effort at training. Perhaps with the introduction of computer teleconferencing into the curriculum, rural Iowa schools can address those areas of greatest concern in benefiting their schools.

References

Lawton, J. & Gershner, V. (1982). A review of the literature on attitudes towards computers and computerized instruction. *Journal of Research and Development in Education*, *16* (1), 50-55.

Lillard, D. (1985). *A survey of Warren County teachers concerning the instructional use of microcomputers*. Paper presented at the Fourth Annual Computers in Education in Maryland Conference, Baltimore, MD.

Moursund, D. (1979). Microcomputers will not solve the computers-in-education problem. *AEDS-Journal*, *13* (1), 31-39.

Nachtigal, P. (Ed.). (1982). Rural education: In search of a better way. Boulder, CO: Westview Press.

Schrum, L. (1988). Introducing teachers to telecommunications: California's ESTTI. *Technological Horizons in Education Journal*, *15* (8), 85.

Sequin, A. (1988). Networking educators across the expanses of Alaska. *Technological Horizons in Education Journal*, *15* (8), 81.

Simonson, M., Maurer, M., Montag-Torardi, M., & Whitacker, M. (1987). Development of a standardized test of computer literacy and a computer anxiety index. *Journal of Educational Computing Research*. *3* (2), 3211-247.

Waggoner, M (1987). *Explicating expert opinion through a computer conferencing Delphi*. Unpublished doctoral dissertation, University of Michigan, Lansing.

Collaboration and Competition: An Innovative Approach to Computer Learning

Ms. Sydney Bennett, Department Head, Business Data Processing
Ms. Judy Kane, Assistant Professor, Business Data Processing
Dr. Pamela Munz, Assistant Dean, General Studies
Nashville State Technical Institute
120 White Bridge Road, Nashville, Tennessee 37209
(615) 353-3406

Abstract

In summer 1989, Nashville State Technical Institute, with supplementary funding, tried an innovative instructional approach designed to help teachers enhance their use of computers in the classroom. Requests from Middle Tennessee teachers prompted the Project Staff to plan a series of summer workshops to help teachers acquire skills that could be applied in their own classroom teaching.

A review of materials suggested an innovative instructional strategy and a model software package that would be suitable for the teachers. A combined collaboration—competition approach was selected as the strategy that would best achieve the learning outcomes desired. For a model software package, the Project Staff selected the Logo language which fit well with the instructional strategy planned.

By placing participants in teams, teachers had the opportunity to collaborate, to share and to form cooperative relationships. Simultaneously, teams engaged in a programming contest with each other, resulting in a positive, inspiring competition to produce a "best" teacher-developed computer program. The results of the teacher projects were stunning. A commendation letter and software package for the winning teachers' schools were sent to their principals which served as an additional incentive.

The Project Staff was in no way prepared for the indepth, sophisticated volume of learning that occurred in each workshop. The teacher projects were creative, and evaluative comments suggested that the atmosphere of friendships and cooperation left the teachers with a feeling of self confidence and a desire to learn more about computers.

Introduction

As society continues to advance technologically, college teaching becomes more complex with new teaching strategies, new objectives and new resources entering the scene. Repeatedly, literature suggests that future education will place greater premiums on effective and efficient educational teaching methods and strategies (Mouton & Blake 1984; Ericksen 1984). More specifically, the goal of two-year colleges, as described by James Hammons in *Changing Instructional Strategies*, is to be a teaching institution existing for the community and responsive to the needs of its citizens. He explains further that in serving these needs, the ultimate goal is more efficient, more effective and less anxiety-producing learning (Hammons 1977).

Considering these educational trends, Nashville State Technical Institute, a two-year college located in the heart of Nashville, Tennessee, recently implemented an innovative program in response to requests by local elementary and high school teachers for additional training in computer skills. With funding from the Education for Economic Security Act, Nashville Tech had additional resources for planning a program of microcomputing seminars to be offered in the summer of 1989 for teachers from seven surrounding Middle Tennessee counties.

Reviewing Materials on Instructional Strategies

Evaluations from previous Logo workshops for teachers indicated the need for a delivery system that would motivate teachers and produce learning outcomes that would be both memorable and worthwhile—learning that the teachers could apply to everyday teaching. In the planning process for the summer program, the Project Staff reviewed a number of sources for guidelines on instructional strategies. In *Teaching Strategies*, G. Ray Musgrave's work on individualized instruction, he explained the benefits of small-group peer teaching in which students work as teams and function as both teacher and student. He suggested that this approach permits students to become partners with the instructor in a cooperative nature with their goals being evaluated by instructor and peers (Musgrave 1975).

In another source, *New Directions for Teaching and Learning*, the author discussed the concepts using the instructor as a facilitator..."unobtrusively guiding rather than controlling the learning process" (Newcombe 1980, p.45). With this approach, the students assume responsibility for their goals, set priorities and problem solve with their peers to achieve success with their projects. Newcombe stated that this approach resulted in students responding more enthusiastically because they could contribute actively and provide support and reinforcement to each other. The student-centered team approach seemed to provide the basis for self-discovery learning through peer support and cooperation.

Like Musgrave and Newcombe, a third source gave similar thoughts on innovative instruction. In "Technology and College Teaching," Christopher

Knapper described the changing trends in instructional strategies. In his discussion of the computer for classroom teaching, he stated:

> ...the fact remains that appropriate use of computers will be an important requirement for many professions in the future. Thus it is not surprising that many universities are struggling to find the best ways of exposing students to a range of relevant computer applications. (Knapper 1988, p. 41)

Knapper concluded that the explosive growth of microcomputers accompanied a tremendous growth of possible applications—and with this a paralleled leap in educational uses. The major implication here, he suggested, would be a greater emphasis on learning skills rather than teaching skills and in the student-centered learning, a focus on generic skills rather than mastery of facts and rote learning (Knapper 1988).

On the issue of student-centered learning, Earnest Boyer, in *College: The Undergraduate Experience in America*, pointed to the need to combine both cooperative learning and *competition* in the classroom to achieve genuine learning outcomes. By doing both, he maintained, competition stimulated ambition and achievement. Cooperation and collaboration then became "musts" in any classroom (Boyer 1987).

What's Best for Computer Learning

Based on the information and conclusions documented in the literature, the Project Staff looked carefully to determine the best delivery system and instructional strategies as well as the best computer software package to use as a model in helping teachers develop computer skills with the least amount of anxiety. For the software model, the Staff selected Terrapin and IBM versions of the Logo programming language, developed at the Massachusetts Institute of Technology in the late 1960s for use in mathematics. Since then, its use expanded to include other disciplines as well. Logo's success with teaching problem solving and creative applications helped make it the Staff's choice for the summer workshops.

The developer of Logo, Seymour Papert, in his book, *Mindstorms*, explains that his research and development of Logo included: "Two major themes—that children can learn to use computers in a masterful way, and that learning to use computers can change the way they learn everything else..." (Papert 1980, p.8). Another writer/teacher explained too, how Logo can enrich the language arts curriculum. "...Students can design and implement animated plays, commercials, stories or announcements..." and in several cases "...create animated maps or timelines: drawing the background with turtle graphics, adding symbols by stamping special turtle shapes, and directing the turtle to conduct a 'guided walking tour' of the significant dates or places displayed on the screen." Use of Logo by teachers helps to break down boundaries between subject areas for students, allowing them to be creative and more oriented toward problem solving (*Classroom Computer Learning* 1986; *Electronic Learning* 1986).

In view of the uniqueness of Logo as a pilot for developing computer skills, the Project Staff looked closely at appropriate instructional strategies that could produce a motivating, exciting and intensive summer training institute for the teachers. To achieve the learning outcomes desired, the Staff chose a peer team approach coupled with the use of a competitive programming contest. The unique twist of student-centered peer teamwork and competitive group approach exemplified Boyer's suggested combination of collaboration and competition as successful instructional strategies. In doing so the teachers would, through the team competition, be motivated to achieve superior results while also benefitting from the support of cooperative peer activities.

The Logo Experience

Participants for the Workshops

Workshop attendees were selected on a first-come basis from a large group of Middle Tennessee teachers who had either attended a previous fundamental computer skills workshop at Nashville Tech or had sufficient verifiable experience with computers to make them computer literate. Although undergraduate college credit provided some incentive for participation, registration and attendance was strictly voluntary. During a five-week period, five different classes provided instruction for twenty participants in each class. The Staff designed the courses for specific grade levels (K-5, 6-8, 9-12) in an attempt to balance interests and abilities for the teams and competition. The intent of this helped to assure equality in the programming contest.

Activities for Learning

At the start of the workshop, it was necessary to provide sufficient basic skills development in the use of Logo to assure that the teachers would be prepared for teamwork activities and individual exploration and learning. To accomplish this, daily activities for the five days provided essential information about the language through lecture and demonstration of command functions using overhead projection with a PC-viewer. Then, short, easily accomplished hands-on exercises reinforced the fundamental concepts and skills necessary to create an incentive for self-discovery and exploration. These activities help create a learning environment exemplary of the self-discovery environment (one of the objectives of the Logo language itself) which could be duplicated in the teacher's own classroom. At the beginning of the classes, the use of team pairs and competition among the teams through the programming contest made it possible to create excitement, anticipation and motivation for the teachers. In addition, the announcement of "prizes" (a software package for the winning teams' schools and a congratulatory letter to each winning teacher's principal) was made on the first day to arouse initial interest.

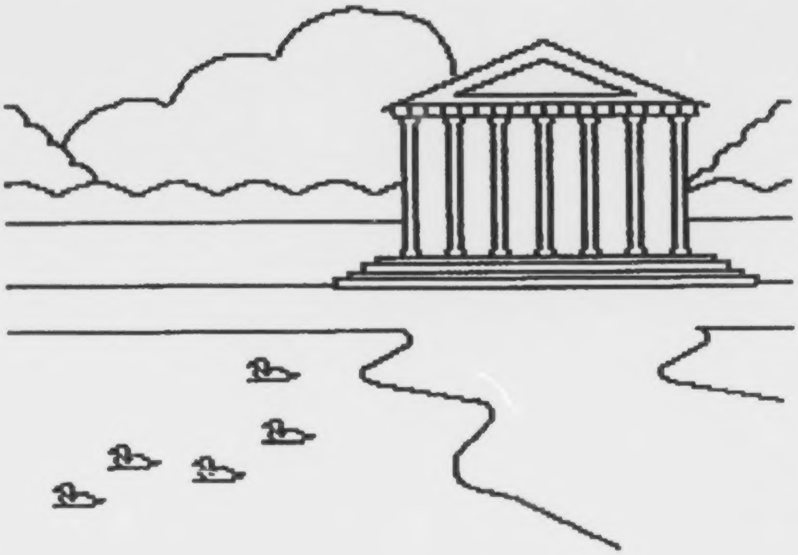


Figure 1. Example Project Result

Initially, the workshop's major objective aimed at providing teachers with the fundamental skills in Logo programming. After two full days of basic skills development, class time provided for creative exploration and idea sharing. Observation of the interaction between the teacher teams and ideas generated by them indicated that the potential creativity and volume of learning far exceeded the Project Staff's initial expectations. As teachers became more involved in the contest, they included in their programs a variety of advanced commands and techniques which resulted in increased competition and more sophisticated learning outcomes. Each day, a few new concepts and skills were presented through lecture and demonstration followed by exploration and guided assistance from the instructor. As the first week of class progressed, it became clear that the instructional strategies chosen were successful and should continue for other later classes.

On the fifth and final day of class, a three-hour block of time was reserved for presentations of the projects by each team to the entire group. A demonstration of the results of the project along with an explanation of the techniques and logic necessary to accomplish the outcomes were required of each team. The peer group then voted for the three best projects which allowed each participant to vote for his or her own project if desired while still expressing an opinion on the two other best projects. A tally of all votes weighted equally determined the winning team.

Subsequent to the first workshop, participants were shown exemplary projects from previous classes which increased competition and added inspiration and self-confidence. Participants from the last four workshops expressed surprise at what had been learned and accomplished by previous teachers in one short week, and many were motivated to try experimentation that would make their projects even more creative. As an example of this, projects from the first week included programs which displayed very accurate and creative graphic designs on the computer screen. (See Figure 1 for an example.) However, after viewing these projects, participants in later workshops began to envision enhancing their own projects to make them better than previous examples by adding animation, flashing lights and text on the screen. They also began to inquire about and discover other capabilities of the language.

Summary

Observations and Learning Outcomes of the Workshops

During the course of each week, motivation and incentive to learn was apparent as many teachers worked through lunch and in the evenings at home on their projects. These activities were in addition to the extensive reading and written assignments required for the coursework. A second observation was the obvious developing friendships which occurred within the workshop teams. The friendships could be seen in the informal exchange and sharing of ideas. Although this had not been a part of the formal workshop planning, the relaxed atmosphere contributed to an enhanced learning environment for all participants.

The contest entries best exemplified the learning outcomes of the computer training. The teachers' entries far exceeded the Project Staff's expectations. The creativity, level of sophistication and volume of learning exhibited by the teachers' Logo programs suggested that the skills acquired over a five-day period were much more advanced and indepth than the staff had predicted.

Evaluation of Workshop Strategy

Many of the teacher participants indicated on the evaluation surveys that the learning approaches used were motivating and inspiring. Comments concerning the instructional strategies included such statements as:

> "I liked the presentations and then letting us explore on the computer"
> "I like the atmosphere for learning"
> "I really enjoyed the contest! Great competition"
> "Contest was motivating"
> "I liked the teamwork approach"

In conclusion, through observations of the workshop activities, assessment of the teacher projects and evaluative comments by the teachers, the Project Staff determined that the cooperative approach (teamwork) used in conjunction with the competitive approach (contest) was an excellent instructional strategy. Through this, the workshops provided a learning atmosphere in which teachers were highly motivated, intensely creative and very successful in acquiring the computer skills they desired.

Selected Sources

Boyer, Ernest L. *College: The Undergraduate Experience in America*. Harper & Row Publishers, 1987.

Ericksen, Stanford C. *The Essence of Good Teaching*. Josey-Bass Publishers, 1984.

Hammons, James O. *Changing Instructional Stratiegies*. Josey-Bass Publishers, 1984.

Knapper, Christopher K. "Technology and College Teaching," *College Teaching and Learning: Preparing for New Commitments*. Josey-Bass, Inc., 1988.

Mouton, Jane Srygley and Blake, Robert R. *Synergogy*. Josey-Bass Publishers, 1984.

Musgrave, G. Ray. *Individualized Instruction: Teaching Strategies Focusing on the Learner*. Allyn and Bacon, Inc., 1975.

Newcombe, Judith P. "The Impacts of Change: A Case Study," *New Directions for Teaching and Learning*. Josey-Bass Inc., 1980

Papert, Seymour. *Mindstorms*. Basic Books, Inc., 1980.

T1-2 MANAGING & FUNDING COMPUTING IN MINORITY INSTITUTIONS (ECMI)

Managing and Funding Computering in Minority Institutions

Jesse Lewis, Chair
Norfolk State University
Norfolk, VA

Abstract

ECMI is an organization run by a steering committee representing institutions whose student body reflects a large identifiable minority population. The objectives of ECMI are to provide services and assistance in: 1) computer literacy; 2) educational computing; 3) research computing; 4) technical assistance-consultants; 5) education programs in the computer sciences; 6) computing facilities; 7) direct student assistance; and 8) the need for a comprehensive program.

This session covered the management and funding of computing at minority institutions. A representative from the Association of Departments of Computer Science and Engineering at Minority Institutions will discuss computing and computer science at minority institutions, and representatives from the National Science Foundation will review funding opportunities, availability, and guidelines.

Panelists

William Lupton
Jackson State University
Jackson, MS

Larry Oliver
National Science Foundation
Washington, DC

Sponsor: ECMI

People, Love, and Computer in Education: A Perspective on Ten Years' Involvement

Betty Collis
University of Twente
The Netherlands

Abstract

Love? At a computers in education conference?

I have been in love a few times in my life, most of these times with a person, but at least once with an idea. That idea was the potential of computers in education. I didn't think of this as love, of course, but now, after more than 10 years of fairly intensive involvement with computers in education, I see some clear parallels. What are these parallels? Why is someone's love-life of any interest to others? It is because I see now so many similarities between what happens, and doesn't happen, with computers in schools, and what happens to people when they are in love, or not in love, that I think there are some important messages for us. And some important questions. So this rather unusual NECC presentation is a personal reflection on what motivates people, both in their relationships with others and in their response to computers in the school.

What are some parallels between being in love and getting involved with computers in education?

I think there are many points to reflect upon, especially for us who are involved in teacher training and in other strategies to promote more extensive use of computers in education. What are some of these? In this abstract I will only mention a few.

- "There is no accounting for chemistry."

I was around computers for many years and felt no interest in them. Then, one day the time was right and I did. This is like love. It's not logic; it's emotional. Can we expect teachers to voluntarily give time, energy, and commitment to computers in education, to have the necessary patience, just because we have logically organized a curriculum about computer literacy for them, if they don't feel something like love?

- What happens when the honeymoon is over?

What sustains the computers in education user after the excitement dies down? We have all seen too many cases of computer coordinator burn-out to not think about this problem.

- Do you have to fall in love to be a computer user in education?

At one level, no, of course not. Many people can see the functional advantage in some aspect of computer use, mainly word processing, and if it is convenient enough, use a computer for certain purposes.

But are functional users the ones who will open up the potential of computers in education, who will revolutionize the system sufficiently so that great ideas associated with computer use (like hypermedia and multimedia and telecommunications-mediated learning activities) can be implemented? No, for these big things to happen, I think we need passion and personal investment. But how long can we expect the committed teacher to keep giving of time and energy, with little or no material reward, such as more time, money, or recognition? Can we count on a new wave of "young lovers" to always be on the scene?

I hope so.

In this presentation, we will look more closely at implications for teacher training and for policy makers. We will ask the question, "How realistic are our expectations and procedures with respect to promoting computer use in schools, based on what we see again and again in human nature?" Can we predict what will bring people to fall in love? Can we keep going if they don't?

Merging Object-Oriented Development Into Introductory Computer Science Courses

Martha McCormick and Ronald White
Department of Mathematical, Computing and Information Sciences
Jacksonville State University
Jacksonville, Alabama 36265

Abstract

Object-oriented development is becoming increasingly important as a method to manage large and complex software systems. This paper presents a strategy of introducing the object-oriented paradigm into introductory computer science courses in order to emphasize software engineering principles. Object-oriented development is presented as a part of the lifecycle in software development. Problems are decomposed based on the concept of objects.

Introduction

Software development has changed dramatically during the past 15 years. The structured approach to development was perfected and accepted and its results were enjoyed. In recent years, object-oriented development has begun to have an influence on software development. It is not a tool to replace the structured approach but to enhance it.

Object-oriented development is presented within the framework of software engineering. Software engineering terms and concepts are introduced and illustrated in CS1 and applied in CS2. The concepts of objects are introduced in the first two courses but not implemented until the student is in the data structures course.

In CS1 emphasis is placed on problem-solving and the development of a technique or system which all students can relate to when faced with the computerization of a problem solution. Departmental experiments continue with different ways to delay the introduction of an implementation language until a solid foundation is achieved in problem analysis and high level design.

Procedural and data *abstractions* are used to entice students to think in terms of the problem environment until a prospective solution can at least be theorized and developed in pseudocode. The "what must be done" analysis is performed while encouraging students to wait until much later to suggest "how to" solutions.

Information hiding is emphasized as students propose alternative algorithms for each operation. Students are encouraged to delay implementation details until an algorithm is selected based on efficiency and whether or not it meets system requirements.

After developing a requirements analysis document, each student must develop a *design document* which outlines the system data flows, major algorithms and modules needed to meet the system requirements. Students are encouraged at this point to develop alternative strategies that will permit *independent modules* to be developed, implemented and tested under control of a driver module. When dealing with algorithms that are difficult to understand, students are encouraged to build *prototyes* that can be used for experimentation and modified into final implementation modules for the project at hand.

For each assignment in CS2, the student must include a *process metrics* and a *product metrics* section in the front-end documentation. Process metrics (in quarter hour units) are required for time spent in analysis and design, time spent in coding and implementation and time spent in testing. Product metrics include lines of code, number of modules and number of global variables used. Although this is an elementary approach, it does give the student an opportunity to see how these metrics change from assignment to assignment.

Data Structures

In the data structures course, the abstract data type is presented as one of the most important constructs. An abstract data type is defined in terms of an object. An object contains the characteristics of an entity and its behavior. By merging these characteristics and behaviors, an object knows everything it needs to perform a task. The characteristics form the interface to the object. The behavior is in the implementation of the object and is hidden.

The student is already comfortable with the ideas of data and procedural abstraction. In the introductory courses, a structured approach to development was used to move from abstraction to implementation. At this point, the fundamental data structures such as stacks, queues, trees, and graphs are presented in parallel with the ideas of object-oriented development. To enhance the understanding of objects, it is useful to have the student think in terms of metaphors. An automobile can be described in physical terms—the number of passengers it will hold, its engine, its exhaust system, and so on. An automobile can also be described in functional terms—it leaves for a destination, it speeds up and slows down, it turns, it reverses, and so on.

The object-oriented development approach is presented in the classic waterfall model: analyze, design, implement units, implement programs, validate and verify. The big difference between the structured approach as opposed to the object-oriented approach is that immediately after the requirements analysis is complete, the object-oriented development focuses upon the data not the process. Object-oriented development is presented in the following manner: identify the abstract object (data) that represents the problem domain, identify the abstract operations supported by the objects (i.e. define the interface to the objects), and the problem solution is simply a sequence of calls to objects.

Turbo Pascal 5.5 is used in the data structures course. It offers two features, *units* and *objects* which can be used to illustrate data abstraction and object-oriented design. Some information hiding may be done with *units* and *objects*. The *unit* is a feature which allows a package of code to be compiled independently. Each *unit* has two parts, the interface and implementation. The interface of the units contain information that is "public" information available to all units that use the unit. It may be composed of a list of the units which it uses; constant, type, and variable declarations; and procedure and function declarations. The implementation section contains the "private" constant, type, and variable declarations used only in the unit. It also contains the bodies of the procedures and functions declared in the interface. There is an optional executable section which contains code that is executed only once before the body of the main program is executed. The *object* is a feature which provides the main properties of a object-oriented programming language:

1. *Encapsulation*: This is the bonding of code and data together. In traditional programming, the physical characterisics (data) of the automobile could be defined as data structure in the following way:

```
Type
   Automobile = Record
      Speed    : data_type;
      Size     : data_type;
      Steering : (left, right);
      Brakes   : (on, off)
   End;
```

The automobile behavior (code) could be defined as procedures and functions.

```
   Procedure Accelerate;
      Begin
         :
      End;
   Procedure Steer_left;
      Begin
         :
      End;
```

In object-oriented programming the characteristics and behaviors are combined into a single entity, an object. The automobile may be defined to look like this:

```
Type
   Automobile = Object
      Speed    : data_type;
      Size     : data_type;
      Steering : (left, right);
      Brakes   : (on, off);
      Procedure Initialize;
      Procedure Accelerate;
      Procedure Decelerate;
      Procedure Steer_Right;
      Procedure Steer_Left;
   End;
```

The *object* contains declarations for both data and procedures. The procedures and functions declared within an *object* are known as methods. The *object* simply defines the method header. The actual code that implements the method is specified separately. Once the *object* is defined, variables are declared by using the object's name.

```
Var
   Sport_car  : Automobile;
```

All actions affecting the object can be made by referring to the object itself.

```
With Sport_Car Do
   Begin
      Initialize;
      Accelerate;
      Steer_Right;
      Decelerate;
   End;
```

Encapsulation is code intensive. Every access to a data field requires a function or procedure call. Turbo Pascal 5.5 does not, however, keep one from accessing an object field directly. It is stressed that accessing an object field directly is both unnecessary and undesirable.

2. *Inheritance*: A type can inherit the characteristics of another type. This means that the descendent acts just like the ancestor type, except for an explicit list of operations that are implemented differently. For example, Figure 1 shows how the object automobile inherits from gasoline-powered vehicle.

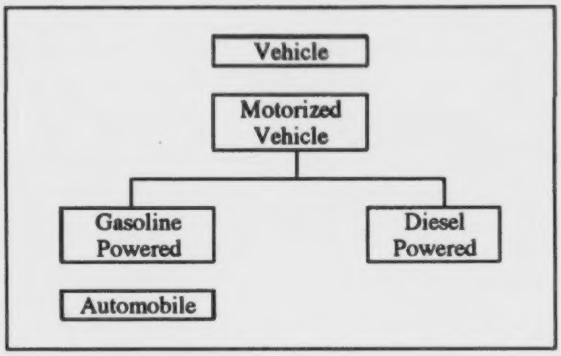


Figure 1. Inherited Data Types

In Turbo 5.5, inheritance is similar to the technique of nesting records.

```
Type
   Movement = Record
      Direction: Data_type;
      MPH     : Data_type;
      Speed_Change: Data_type
   End;
   Status = Record
      Mvmt = Movement;
      Mileage = Data_type;
      Gasoline = Data_type
   End;
```

Object inheritance replaces the need for nested procedures. The following shows how to use objects to replace the nested record declarations above:

```
Type
   Movement = Object
      Direction: Data_type;
      MPH     : Data_type;
      Speed_Change: Data_type;
      Procedure Initialize
   End;
   Status = Object (Movement)
      Mileage: Data_type;
      Gasoline: Data_type;
      Procedure Initialize
   End;
```

The statement Status = Object (Movement) means Status inherits everything contained in Movement.

3. *Polymorphism*: In object-oriented programming polymorphism refers to the ability of a method to perform in various ways, depending upon the structure of the calling element. An *object* is structured such that when called, it can determine what it is being asked to perform and will act accordingly.

   Object variables in Turbo Pascal 5.5 follow slightly different compatability rules than do normal Turbo Pascal variables. The primary difference is that an ancestor type is compatible with a descendent type, but the reverse is not true. The flexibility of object type compatability make polymorphism possible. This means a procedure may accept a wide range of object types, even if it is unaware of them at compile time.

   Polymorphism has another implication: if a procedure accepts a polymorphic variable at run time, compatible objects may be defined without recompiling the unit that contains the procedure. This means one does not have to think of everything in advance.

The use of the object-oriented paradigm and the Turbo Pascal 5.5 compiler provides the tools the instructor needs to introduce the concept of *reusable code*. We feel this should be part of any data structures course.

The projects that are assigned teach the student to make use of existing units to construct a software system. They are penalized for coding from the ground up when they have units available that perform the same function. The introduction of procedures and functions was viewed as a first step in software reuse. They were presented as reusable modules within a program. Turbo Pascal 5.5's unit and object are presented as a more advanced step of reuse. Building black boxes is presented as the essence of object-oriented development. The student is taught to focus on the data in the project and to determine the appropriate encapsulations. These stand-alone objects or black boxes may be used to assemble any number of systems.

Evaluations/Conclusions

It has been difficult to quantify success, probably because the benefits have been more intangible than measurable. However, it is believed that most students develop a "software engineering attitude" that will serve them well in future courses and in their resulting careers.

The following evaluations are based on observation and from student evaluation comments.

1. The software engineering approach has increased the student's awareness of the need to design "in the large".
2. The student's exposure to data abstraction and object-oriented methodology has helped them to

understand and appreciate the concept of reusable code.

3. The students seem to have a better understanding of data types and the idea of developing custom data types to meet specific program needs.
4. The object-oriented approach to development offers the student a tool to better capture a model of the real world.

Overall, students seem to appreciate the approach we take in teaching software development. Poor overall performance is exhibited by some students, probably due to their failure to follow guidelines given in developing the solution.

References

1. Bailey, S., Designing Objects. *Computer Language*, Jan. (1989), 34-43.
2. Booch, G., Object-Oriented Development. *IEEE Transactions on Software Engineering*, SE-12, (1986) 211-221.
3. Booch G. *Software Engineering with Ada*, Benjamin Cummings, Menlo Park, CA, 1986.
4. Brackett, J., Software Requirements. *SEI Curriculum Module SEI-CM-19-1.0*, Carnegie-Mellon University Software Engineering Institute, 1988.
5. Cox, B. *Object Oriented Programming, An Evolutionary Approach*, Addison-Wesley, 1987.
6. Gilbert, P. *Software Design and Development*, Science Research Associates, Chicago, IL, 1983.
7. Grady, R. and Caswell D. *Software Metrics: Establishing a Company-Wide Program*, Prentice-Hall, Englewood Cliffs, NJ, 1987.
8. Liskov, B. and Zilles, S., Specification Techniques for Data Abstactions. *IEEE Transactions on Software Engineering*, SE-1, 1, (1975) 7-19.
9. Liss, I. and McMillan, T., An Example Illustrating Modularity, Abstraction & Information Hiding Using Turbo Pascal 4.0. *SIGCSE Bulletin, 21*, 1, (1989) 93-97.
10. Mills, E., Software Metrics. *SEI Curriculum Module SEI-CM-12-1.1*, Carnegie-Mellon University Software Engineering Institute, 1988.
11. Morrell, L., Unit Testing and Analysis. *SEI Curriculum Module SEI-CM-9-1.1*, Carnegie-Mellon University Software Engineering Institute, 1988.
12. Pugh, J., LaLonde, W., and Thomas, D., Introducing Object-Oriented Programming into the Computer Science Curriculum. *SIGCSE Bulletin, 1* (1987) 98-102.
13. Stubbs, D. and Webre, N. *Data Structures with Abstract Data Types and Pascal. Second Edition*, Brooks/Cole, Monterey, California. 1989.
14. *Turbo Pascal 5.5 OOP Guide*, Borland International, 1989.

Historical Perturbation and Reconstruction: A Computers and Society Teaching Method

William J. Joel
Division of Computer Science & Mathematics
Marist College
Poughkeepsie, NY 12601
Email: jzem@marist.bitnet

Abstract

A method has been developed to aid students in their understanding of the historical impact of computer technology. This method consists of presenting a student with an historical environment, perturbing one or more events, having the student determine the effect of these perturbations and asking the student to compare known consequences with theoretical end results.

Introduction

One of the objectives of the course Computers and Society, as taught at Marist College, is to demonstrate to the student the impact of computer technology from an historical perspective. In order to afford students an understanding of the historical effect of computer technology, a method was developed for studying those effects.

Method

Description

There are essentially four steps to this method:

- presenting a student with a scenario describing what is essentially a commonly known use of computer technology in history,
- perturbing an historical event by either changing, adding or eliminating it,
- asking the student to reconstruct this historical path to the present, based upon the given perturbation, and
- having the student compare this new end result with the known historical consequences.

In addition, the student is further encouraged to diverge from this main line of reconstruction whenever it is apparent that other historical paths are affected by the suggested perturbation. The purpose of this additional step is to demonstrate, to the student, that no historical event occurs in a vacuum. That is, many historical paths cross each other, and thereby affect each other's progress.

The student must have an ample background in order to perform the required task, in that this method

> ...requires that the student: (a) place in cognitive perspective the multiple forces operating upon history at the time of redirection; (b) assess both the status and the dynamics of each of these forces, independent of the single element or force which is to be redirected or eliminated from the historical constellation; (c) determine the nature of the interaction between the continuing forces and the force to be redirected or eliminated; (d) formulate, from this knowledge and from other experience, generalizations about probabilities of cause and effect in these spheres; (e) and finally to rewrite or re-create subsequent history through the application of these generalizations about historical cause and effect. (Dumas, 1969)

Discussion

Historical vs. future extrapolation

The reader may ask why extrapolation from a point in history as opposed to extrapolation into the future? There are essentially three reasons:

- The first, and possibly most obvious, reason is that extrapolation from the past to the present is simpler. It is assumed that the student already has a basic grasp of the historical environment under examination. The knowledge of existing historical patterns allows the student to have a framework, or template, with which to construct this extrapolation. Going into the future is more difficult in that there is no template to follow. One should learn from history, but the future has no history yet to base theories upon.
- Second, the course Computers and Society is a one semester course and generally covers a wide gamut of sociological effects, utilizing a variety of teaching methods. Therefore, a method that is less time consuming is preferred.
- A third reason for historical reconstruction of the past is that it forces the student to first have as complete an understanding of the basic historical environment as possible. Students must be able to differentiate between cause-effect chronologies and simple sequences of unrelated events. The former will have a direct input to the student's reconstruction.

If a student has difficulty in performing the assigned reconstruction, then a possible cause would be insufficient knowledge of the related history. The student will therefore need to fill in these gaps before proceeding with the reconstruction.

Examples

During the last few years, many exercises have been developed based upon this method. Several of these scenarios are described below. First, for each exercise a scenario is stated describing the specific historical area and perturbation. Then, comments are made as to how a student might and should go about historical reconstruction.

Newspapers

Scenario:

The use of computers for word processing, page layout and platemaking could not be sufficiently developed by the newspaper industry due to extreme difficulties in implementing the technologies.

Comments:

Note how only the newspaper industry has been singled out. This implies that all other print industries did implement this technology successfully. An obvious effect would be the demise of newspapers as we know them or a radical metamorphose into something new. But this obvious effect could be misleading.

The student should ask, at this point, why did newspapers have such difficulty? After developing a hypothesis, the student should see if these proposed causes could have affected any other print industries. Again, no historical event occurs in a vacuum.

After a set of causes has been determined, the student must trace the effect trail for this perturbation to the present. A possible final effect would be the growth of teletext as a replacement for the weekly/daily news coverage newspapers gave.

Space exploration

Scenario:

It was decided, by NASA, that for public relations reasons exploration into our solar system would be by manned space probes only.

Comments:

One might quickly observe how on the surface this has nothing to do with computer technology, but then that would be hasty. Much of space exploration is based upon computer control of processes such as launch and landing sequences, life support systems and telemetry. To state that only manned probes will be used implies a shift in the emphasis as to how computer technology is used.

The student should ask several questions. If computers are used for life support, were they sufficiently advanced for life support in long distance trips? To support a manned probe to the moon is quite different than a manned probe to Mars.

As there is a sizeable time-delay to send and receive messages from a probe in space, were computers fast enough to perform needed flight calculations either on-board or from Earth, to offset this delay?

What the student might come away with is that computer technology was not sufficiently advanced in the 1950's or 60's to adequately support manned space probes, and that such exploration could have been delayed. An alternate conclusion could have been that computer development was accelerated as a result of this decision, noting how space exploration has acted as an exciter for many other technologies.

Education

Scenario:

Instead of donating computers to schools, many vendors simply sold them at a slightly reduced cost.

Comments:

The first question a student should raise is how this would have affected the introcution of computers into the classroom. One possible theory might be that only those few school districts that could afford computers would have bought them, and that the majority of the schools would have either bought one or none.

School districts might have then put pressure on government agencies to subsidize the purchase of computers for schools. Other funding sources might have been explored to take the place of computer donations.

Tangent industries, such as software publishers, would have also suffered. With a much lower number of computers in schools, much less software packages would have been sold.

A final question a student might pose is whether this perturbation would have any affect on whether or not schools would utilize computers in the classroom. Note that this question asks if it would occur, but not when. This distinction is very important and should not be taken too lightly.

Cognitive skills

Any teaching method should re-enforce several cognitive skills. Those skills that are addressed by this method include extrapolation, analysis of relationships and production of a set of abstract relations.

Knowledge

In order for the student to be able to perform the requested reconstruction, an understanding of the basic historical environment must be present. This method therefore strengthens a student's knowledge of the basic historical facts.

Comprehension

The major skill that is emphasized in this method is the cognitive skill of extrapolation.

> [The student] must be able to extend the trends or tendencies beyond the given data and findings...to determine implications, consequences, corollaries, effects, etc., which are in accordance with the conditions as literally described in the original communication. (Bloom, 1956a)

Analysis

Relationships, between various historical paths, are explored by this method. It is important that the student be able to see not only the historical significance of computer technology as a microcosm of history, but also in the grand scheme of history.

Synthesis

The act of perturbing an historical event and subsequent reconstruction not only asks the student to see whether actual subsequent events will still occur, but possibly to synthesize additional, new events.

> [The] student begins with some basic propositions or other symbolic representations and...must deduce other propositions or relations. (Bloom, 1956b)

Evaluation

Once the reconstruction has been completed, the student should be able to determine if the reconstruction is valid. This can be accomplished by comparing it to the existing historical path to see how the new path differs, if at all.

Conclusions

In conclusion, the ability of a student to perceive the historical impact of computer technology is greatly enhanced by the reconstruction of an historical path based upon a perturbation of event(s). To merely study history can be a dry experience, but to reach in and trace its development, with a twist as suggested, can be not only informative but also fun.

It must be remembered that the teaching of any course requires the interweaving of many methods with the end goal of stimulating a student's learning. Learning need not and should not be dull.

> The student is hereby offered an opportunity to do something actively with history. He is challenged to re-create hypothetically both history and the present, in order that we may more confidently formulate generalizations with predictive value which will enable us to create a better society for the future—rather than disastrously 'adjusting' to the future which chaotic events offer us. (Dumas, 1969)

Learning should be as exciting and enticing as possible. It is felt that the method described above fills boths of these needs.

References

Bloom, B. S. (1956a). *Taxonomy of Educational Objectives (Handbook 1: Cognitive Domain)* (p. 95). New York: David McKay Co., Inc.

Bloom, B. S. (1956b). *Ibid*, (p. 171).

Dumas, W. (1969). Speculative Reconstruction of History: A New Perspective on an Old Idea. *Social Education*, *33*, 54-55.

Thinking Skills, Databases, and Nutrition

Gertrude W. Abramson
City University of New York
35 Highview Road
Suffern, New York 10901
(914) 357-2765

Abstract

The computer is a ubiquitous tool in the instructional process; with appropriate software, it can enhance teaching and learning across the curriculum. A major objective of a graduate, computer-education course in Spring 1989, was to develop a prototype project for the use of databases in junior and senior high school classrooms. A database on the nutritional value of foods was the unanimous choice for model because of its easily delineated parameters, its applicability in different subjects, and the high level of interest in physical fitness today. Modifications of the lesson described below are currently being implemented to meet widely divergent curriculum objectives such as solving word problems (mathematics), performing measurements (science), and preparing balanced diets (health education).

Introduction

When teachers plan for instruction they must observe prescribed learning objectives even when there is evidence that different activities would better serve student needs. Instruction that brings about effective learning begins with gaining the attention of the learner (Gagne, Briggs and Wager, 1988). By carefully integrating computing experiences into the existing curriculum, it is possible to retain student interest and enthusiasm and provide serendipitous opportunities for critical thinking, analysis and synthesis. This paper discusses a project-in-process: identifying content for database organization, creating and using the database, linking the database to other instructional materials to maximize learning, and implementing the model in different classrooms.

Learning to teach with databases

A database is an organized way to keep track of a great deal of related data. An electronic database is a tool that facilitates the storage of data within a structure composed of fields, records and files. This data may then be retrieved, modified, sorted and printed (Sachs & Kronstadt, 1989; Arnston, 1988; Ingalsbe, 1987). Opportunities to incorporate this tool into the thinking and learning processes are limited only by the creativity of the teacher and the students.

AppleWorks, PFS First Choice, and Microsoft Works are integrated software programs readily available in schools (Abramson, 1989). Students who have keyboarding skills and have used the word processor of the integrated environment need little additional computer learning time to effectively use the database tool. The new learning relates to the columnar organization of data storage, file management concepts and vocabulary. Introducing students to database concepts through the manipulation of existing databases will often help them understand the underlying processes and whet their interest in database construction (Roberts, Carter, Friel & Miller, 1988).

The class was comprised of teachers from grades four through junior college, 11 female and 5 male, and a female instructor. Database use was introducted by searching ERIC on CD-ROM and by using a commerical social studies database with PFS File. Working cooperatively in groups of four, sample databases involving allowance budgeting, baseball statistics, presidents of the United States, and the animal kingdom were constructed, sorted and searched and short reports were printed. It took about three, 150 minute sessions for the teachers to agree that tool mastery was an easy process.

The next step was to compile a model database that some of the teachers would use in their classrooms the following semester (as part of another course). In search of a subject for the model, the class reviewed computer-education literature and curriculum guides. Guidelines for the search process were:

1. the subject must interest male and female students,
2. the subject must "fit" many curriculum objectives,
3. the subject must provide thinking opportunities,
4. the computer must be a good medium for instruction.

Following lively discussion, agreement was reached that everyone is interested in food and is constantly bombarded by a never ending flow of advice communicated through the popular media. A number of health and fitness books were perused and Stein (1987) was selected as the data source because of its clarity and simplicity.

The data structure was decided upon cooperatively and each of the four groups entered data for different food groups so that precious learning time would not be spent in data entry. The data files were then appended to one another. Care was taken to limit the number of fields to eight so that entire records could be printed on standard printer paper using compressed print.

Nutritional Value of Foods

Foodgroup	Subgroup	Food	Portion	Calories	Protein(G)	Carbohydrates(G)	Fat(G)
Beverages	Alcoholic	Wine, dry	3.5 oz.	85	0.0	4.0	0.0
		Wine, sweet	3.5 oz.	167	0.0	12.0	0.0
	Carbonated	Seltzer	12 oz.	0.0	0.0	0.0	0.0
		Pepsi cola	12 oz.	156	0.0	39.4	0.0
Dairy	Cheese	Mozzarella (skim)	1 oz.	80	8.0	1.0	5.0
		Cream Cheese	1 oz.	100	2.0	1.0	10.0
		Cottage cheese 4.2%	3 oz.	90	12.0	2.0	4.0
		Cottage cheese 1%	3 oz.	61	10.0	2.0	1.0
		Mozzarella (whole)	1 oz.	90	6.0	1.0	7.0
		Ricotta (whole)	1 cup	430	28.0	7.0	32.0
		Ricotta (skim)	1 cup	340	28.0	13.0	19.0
	Miscellaneous	Milk 1%	1 cup	100	8.0	12.0	3.0
		Ice milk 4.3%	1 cup	185	5.0	29.0	6.0
		Ice cream 16%	1 cup	350	4.0	32.0	24.0
		Ice cream 11%	1 cup	270	5.0	32.0	14.0
		Milk, whole	1 cup	150	8.0	11.0	8.0
		Yogurt, fruit	8 oz.	230	10.0	42.0	3.0
		Yogurt, plain	8 oz.	140	12.0	16.0	4.0
Fish	Broiled	Herring	3 oz.	217	20.8	8.0	4.0
		Halibut	3 oz.	147	21.5	0.0	6.0
		Bluefish	0.5 fish	192	32.0	0.0	6.3
		Salmon	3 oz.	156	23.0	0.0	6.3
	Canned	Tuna, oil	3 oz.	170	24.0	0.0	7.0
		Tuna, water	3 oz.	127	28.0	0.0	0.8
		Salmon, pink	3 oz.	120	17.0	0.0	5.0
Fruit	Fresh	Apple	2¾ in.	80	0.0	20.0	1.0
		Banana	1 med.	100	1.0	26.0	0.0
		Cantalope	½ small	80	2.0	20.0	0.0
	Juice	Grape	1 cup	165	1.0	42.0	0.0
		Grapefruit	1 cup	100	1.0	24.0	0.0
		Orange	1 cup	110	2.0	26.0	0.0

Figure 1. Nutritional Value of Foods

After editing was complete, everyone was given a paper copy of the database as shown in Figure 1. A list of activities-cum-questions were developed and cooperatively explored and expanded. For example,

1. A lesson in logical thinking: Are the data listed in a form that will allow for easy menu planning with clear-cut substitute foods? If yes, what evidence do you have to support your conclusion? If no, what do you think needs to be done?
2. A lesson in mathematical story problems: Mary is very hungry and is is also trying to lose weight. She has decided to treat herself to one cup of something sweet. What are her choices? Compare one cup of fresh strawberries to one cup of ice cream.
3. A lesson in measurement conversion: A determination was made that equivalent portions and consistent measures are essential for constructing comparable and contrastable menus. Use search and replace to change all (portion size = one cup) to (portion size = 8 ounces).

Two of the reports generated during the exploration process are shown below. The first is a report of all foods belonging to a particular foodgroup which is shown in Figure 2.

The second used an operator, (calories > 250), and printed only selected fields as shown in Figure 3.

Reflections on teaching with databases

During experimentation time, instructional media were gathered and used with the database program in the different subject areas. Some of these materials were manipulatives and others were print: menus from fast food stores and restaurants, books on diet and fitness, nutrition charts and textbook chapters, kitchen and diet scales, measuring spoons and cups, and different size plastic cups. Very quickly, the nutrition database became one more tool in the learning process and discussion focused on finding enough time to do all the exciting lesson components suggested.

It is easy to get carried away with the fun and elegance of working with computers. The role of the teacher is paramount in the learning process and computing activitiesare great tools to enhance lesson development and implementation. Rules for working with databases are the same as for any instructional medium (Abramson, 1987):

1. the teacher must be comfortable with the medium,
2. learning outcomes must be specified at the beginning,
3. ways of assessing mastery must be devised and used,
4. the teacher must encourage thinking and experimentation.

It was anticipated that three of the teachers would implement the model in their classrooms during Fall 1989 and would collect pre and post-test data regarding subject matter mastery. Unfortunately, the college cancelled the graduate, educational computing, follow-up course. Variations of the model were used in a junior high school resource room to solve mathematical word problems and in a high school health education class to help prepare balanced diets. Although no data was collected, both teachers reported an unusually high interest level among the students, a decreased absence rate during the unit, and test scores that were at least as good as the previous year's when no time was spent on database concepts.

Dairy Foods					
Food	**Portion**	**Calories**	**Protein(G)**	**Carbohydrates(G)**	**Fat(G)**
Cottage cheese 1%	3 oz.	61	10.0	2.0	1.0
Cottage cheese 4.2%	3 oz.	90	12.0	2.0	4.0
Cream Cheese	1 oz.	100	2.0	1.0	10.0
Ice cream 16%	1 cup	350	4.0	32.0	24.0
Ice cream 11%	1 cup	270	5.0	32.0	14.0
Ice milk 4.3%	1 cup	185	5.0	29.0	6.0
Milk 1%	1 cup	100	8.0	12.0	3.0
Milk, whole	1 cup	150	8.0	11.0	8.0
Mozzarella (skim)	1 oz.	80	8.0	1.0	5.0
Mozzarella (whole)	1 oz.	90	6.0	1.0	7.0
Ricotta (skim)	1 cup	340	28.0	13.0	19.0
Ricotta (whole)	1 cup	430	28.0	7.0	32.0
Yogurt, fruit	8 oz.	230	10.0	42.0	3.0
Yogurt, plain	8 oz.	140	12.0	16.0	4.0

Figure 2. Dairy Foods

High Calorie Foods			
Subgroup	**Food**	**Portion**	**Calories**
Cheese	Ricotta (skim)	1 cup	340
	Ricotta (whole)	1 cup	430
Miscellaneous	Ice cream 16%	1 cup	350
	Ice cream 11%	1 cup	270

Figure 3. High Calorie Foods

References

Abramson, G. (1987). Planning for computer-enhanced instruction. *Proceedings of NECC'87*, National Educational Computing Conference.

Abramson, G. (1989). Math in our lives: with spreadsheets. *Proceedings of NECC'89*, National Educational Computing Conference.

Arnston, J. (1988). *dBase III Plus Concepts, Exercises and Applications*. CA: Ashton-Tate.

Gagne, R., Briggs, L. & Wager, W. (1988). *Principles of Instructional Design*. NY: Holt, Rinehart, Winston.

Ingalsbe, L. (1987). *dBase III and dBase III Plus for the IBM-PC*. OH: Merrill.

Roberts, N., Carter, R., Friel, S. & Miller, M. (1988). *Integrating Computers into the Elementary and Middle School*. NJ: Prentice-Hall.

Sachs, D. & Kronstadt, B. (1989). *Discovering Microsoft Works for the IBM Personal Computer*. NY: Wiley & Sons.

Stein, L. (1986). *The Bloomingdale's Eat Healthy Diet*. NY: St. Martin's Press.

Computer Use for Newswriting Evaluation and Feedback

William Edward Smith
Assistant Professor, School of Journalism
Northeastern University
102 Lake Hall
Boston, MA 02115
(617) 437-4050 Internet: billsmith@Lynx.northeastern.edu

Abstract

This research was designed to determine whether computer based, individually-tailored feedback on writing content and style is more effective in prompting successful rewriting by introductory level newswriting students than generic feedback presented in paper handout form. Sixty-seven college students participated in the study. The performance by the computer group was significantly better, though there was a substantial gender-based discrepancy in scores. Male students did much better using the computer, while female students did slightly better working on paper. A follow-up survey of a subset of the students disclosed that the males had more experience with computers and more favorable attitudes toward the machines, suggesting that as female students become more experienced computer users their scores should improve as well.

Introduction

Computers have gained widespread acceptance in journalism schools as a replacement for the typewriter in newswriting courses, but journalism educators have developed "relatively little specialized software."[1] A journalism professor published what's believed to be the first program designed to simulate the process of gathering information for a news story in 1984.[2] But apparently less than a half dozen additional examples of the genre have appeared.[3]

This paper reports on a test of the effectiveness of a newer type of program—a reporting simulation that also provides help with writing. These programs attempt to supplement the vital role of the instructor in a newswriting class by adding artificial intelligence features to the reporting simulation. The added features permit a student reporter to use the information gathered in the simulation to write a lead paragraph for a news story—the opening paragraph that traditionally summarizes the most important elements of what has happened. Journalists generally consider the lead to be "the most important part of a story—and the most difficult part to write."[4] It sets the tone for the story and makes a promise to the reader about what additional information the rest of story will deliver. Newswriting texts generally devote much of their space to the special issues involved in lead writing.

In the computer program, after writing a lead, the student can "call the editor" for a critique. After that, the student rewrites the lead based on the critique and calls the editor again for an assessment of the revised draft.[5]

In part this procedure simply automates and makes more immediate the grading process performed by the instructor. It also builds into the process not only the opportunity to rewrite—but the chance to get meaningful feedback on the revision as well. Even the most conscientious instructors frequently find they lack time to critique rewrites as thoroughly as they do the first version of a story. The built-in encouragement and rewarding of revision also fits in with current theories in English composition that stress a "process approach" to writing.[6]

The computer critique has its limits. At this stage of the development of such programs, the computer is not able to catch all errors—even in a passage as short as a paragraph and even with story-specific rules about appropriate content. In addition, it occasionally sees an error where none exists. But it does catch a large number of the most common errors, and it never wearies of pointing them out.

---

[1] Gale Wiley, "ANPA-Texas Clearinghouse Evaluates Teaching Software," *Journalism Educator*, 43(3):39-40 (Autumn 1988).

[2] Peter Owens, *Super Scoop: Instructor's Manual* (Wentworth, NH: COMPress, A Division of Wadsworth, Inc., 1984).

[3] The Clearinghouse for Computer-based Education in Journalism and Mass Communication, Department of Journalism, University of Texas-Austin, in February 1989 listed the following: *City Council* (1988) by Richard Cameron, for the Apple Macintosh; *Fire—Reporting Simulation Game* (1986) and *Fatal* (1987) by John V. Pavlik, for the IBM-PC; *Super Scoop I* and *Super Scoop II* by Peter Owens, for the Apple II.

[4] Fred Fedler, *Reporting for the Print Media*, (New York: Harcourt Brace Jovanovich, 1989), p. 73.

[5] Part of the decision-making process the computer assists the student in mastering while writing the lead is selection and organization of the story elements that will go into it. Edward B. Versluis reports on several entertaining essay-structuring simulation programs designed for a similar task in English classes in "Computer Simulations and the Far Reaches of Computer-Assisted Instruction," *Computers and the Humanities*, 18:225-232 (1984).

[6] Philip D. Gillis reports on the use of computer-aided instruction techniques at an earlier stage of the writing process in "Using Computer Technology to Teach and Evaluate Prewriting," *Computers and the Humanities*, 21:3-19 (1987).

So, it is not a replacement for the instructor's more extensive knowledge of the writer's craft. But it seems to serve as a helpful assistant. It gives students additional practice without additional grading burden on the instructor, and it frees the instructor to deal with the more subtle and interesting questions of writing style while the computer focuses on the routine and basic.

Further development of such programs may lead to creation of more general models of the news story that students could explore. Programs already exist that model generic forms of gossip[7] and poetry.[8] Research targeted at college-level remedial-writing instruction is also underway to develop programs that would recognize sentence-level errors in any student text.[9]

The remedial-writing researchers argue that "Students can best be taught to edit their writing for errors if they can receive instruction on the spot."[10] Reporting simulation programs with writing analysis features provide that immediate feedback—whether used in the classroom or as a homework assignment.

Two reporting simulation programs have been developed with the newswriting analysis feature, one for the IBM PC[11] and one for the Apple Macintosh.[12] This test was limited to the Macintosh program, Bayshore Blast, because the journalism school where the test was conducted has a Macintosh lab but not an IBM PC lab.

Research Question

The study was designed to answer the following research question:

> Are pre-packaged, individually-tailored comments delivered by computer more effective in helping students rewrite their leads than generic feedback with similar content delivered on paper?

In addition, the data was analyzed to detect any gender-based differences in the results.

Methodology

Sixty-seven students in six different sections of introductory newswriting classes at the researcher's university participated in the study.

Students included in the experimental group ranged from freshmen to graduate students. Some sections completed the exercise very early in the term, others toward end of the term. At the option of the instructor of each section, students in a section were either all assigned to the computer version, all assigned to the paper version, or permitted to choose which version they wished to complete. Given the sample size, it was not possible to control for these three variables and produce significant results, so the results are aggregated for this report. Thirty-six students took the computer version of the exercise, 31 the paper version.

In the computer version, students chose the locations to which they'd travel in search of interviews, the people they'd interview and the questions they'd ask—all from a series of menus. They had an electronic note pad available on which to type notes from the interviews. They also could go back to re-interview someone or interview additional sources after starting to write the story, but informal observation of the class sessions indicates few students did so. These students also wrote their lead on the computer and then pressed a button to "Call the Editor" and get the feedback. After reading over the editor's comments, they rewrote their leads. The students could then repeat the process of calling the editor for comments and rewriting until the 65-minute class period ended.

In the paper version students had the same amount of time to complete the assignment. At the start of class they received a six-page handout, which contained transcripts of all the interviews contained in the computer version, along with additional descriptive information presented on the computer screens. They could take separate notes or mark the handout as they wished as they read through it. These students then typed (or in a few cases hand wrote) a lead paragraph for the story. When they informed the instructor that they had completed their best-effort first version, they were given a two-page handout entitled "Confidential Memo on What Your Editor Wants in the Lead" and told to use it in preparing a rewrite.

There is a methodological question about the use of the handout as the feedback in the control version of the exercise. Such standardized written feedback is not the norm in journalism classes. Generally instructors collect student papers, mark them up individually and then return them in time for the next class session. This process has the advantage of making all the instructor's knowledge available for the critique, but it provides no assurance of consistency in the critique given to each paper in the class. The desire to assure consistency and to provide immediate feedback led to adoption of the paper handout as the control group procedure.

---

[7] E. Paul Goldenberg, "Linguistics, Science, and Mathematics for Pre-College Students: A Computational Modeling Approach," *Proceedings, National Educational Computing Conference*, June 1989, Boston.

[8] Two poetry-generating programs are described in: Helen Schwartz, "Monsters and Mentors: Computer Applications for Humanistic Education," *College English*, 44:145-46 (1982).

[9] Glynda Hull, et. al., "Computer Detection of Errors in Natural Language Texts: Some Research on Pattern-Matching," *Computers and the Humanities*, 21:103-118 (1987).

[10] Hull, *ibid.*, at p. 116.

[11] John V. Pavlik, *Fatal v. 2.0*, (Austin, TX: Wayne Danielson Software, 1988).

[12] William E. Smith, *Bayshore Blast: a reporting simulation with newswriting analysis*, forthcoming from Duke University Press.

For both groups of students the feedback was divided into six categories:

1. Proper handling of the time element—when the story happened. (Since the students were writing today for publication in tomorrow's paper, there was potential for substantial confusion among novice newswriters about this issue.)
2. Description of the main event that was the focus of the story.
3. Correct reference to the location of the event.
4. A brief summary of the major consequences of the event.
5. Avoiding passive voice constructions in the lead.
6. Properly applying an Associated Press Stylebook rule commonly disregarded by novice newswriters.

The different categories were assigned values ranging from four to eight points and most categories were subdivided so that a marginally adequate response might get one or two points and the copy would have to be right on target to get the maximum points for the category. The maximum score for all six categories combined was 30 points.

Examples of the paper and computer versions of the feedback for one of the six categories are provided in Figures 1 and 2.

The computer program recorded the first and final version scores received by each student who used the computer version. The paper versions were later run through the computer program for scoring.

Results

Two students who experienced software-incompatibility start-up difficulties with their computers

Paper Version Feedback

For first of the six feedback categories -- Time Element -- students receive all the following comments:

1 -- Time Element

General rule:

Your editor wants to see a reference to when the event happened.

Basic Application:

Since you are writing a story about an event today that will appear in tomorrow's paper, the time reference should be "yesterday."

Fine-Tuning:

A. Since the editor always thinks people are impressed by inclusion of fresh news in the paper -- events that happened relatively close to deadline -- she'll particularly like it if you say "yesterday afternoon."

B. The time element is rarely the most important item in the lead, and many stories on any given day will have the same time element, so to avoid over-emphasizing it and creating a repetitiveness problem, don't place the time element in the first few words of the lead.

Figure 1.

Computer Version Feedback

For first of the six feedback categories -- Time Element -- students receive one of the following comments:

If the student fails to mention "today" or "yesterday", then the computer says:

TIME ELEMENT: When did this story happen? You seem to have forgotten to say that, or miscalculated it. (0 points)

Else, if the student says "today", then the computer says:

TIME ELEMENT: You seem to have gotten it wrong. (0 points)

Else, if the student says "yesterday" but puts it in the first 45 characters of the lead, the computer says:

TIME ELEMENT: Good-- you mentioned it. But WHAT happened is more important than WHEN, so put the time element later in the lead. (2 points)

Else, if the student says "yesterday" but doesn't say "afternoon", the computer says:

TIME ELEMENT: OK, but add 'afternoon' to make it better. (3 points)

Else, the computer says:

Good, (name), you got the time element just right. (4 points)

Figure 2.

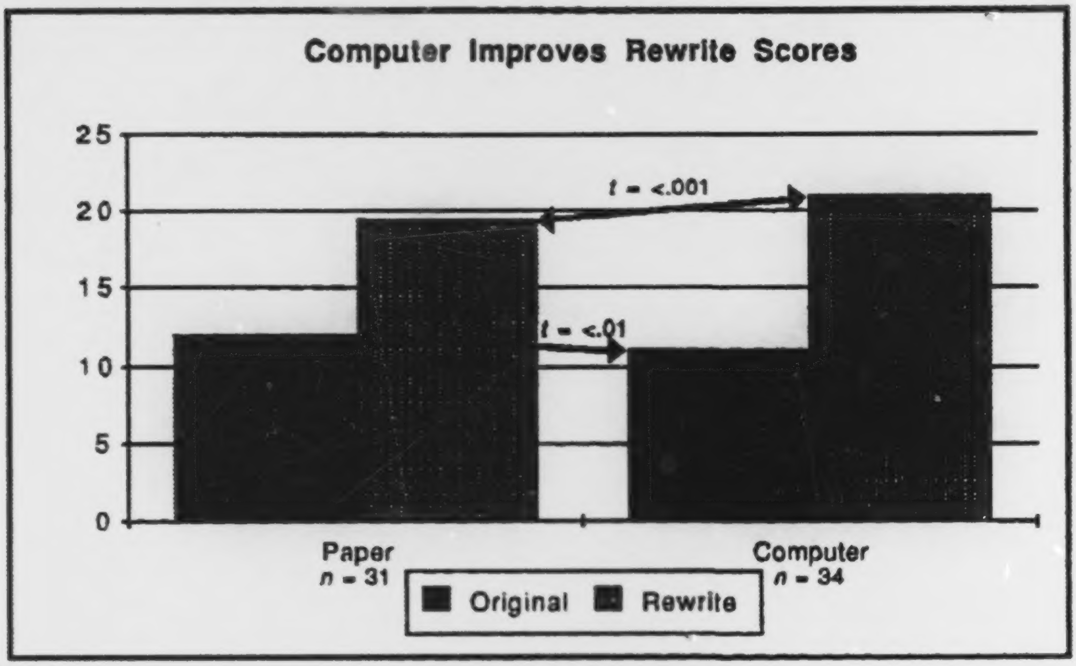


Chart 1.

and didn't get far enough into the exercise to write even a first version of the lead had to be excluded from the sample. For the rest, the computer proved to be more effective in presenting the rewrite suggestions. The mean final scores were 21 for the computer users and 19.5 for the paper handout group. Application of a two-tailed *t*-test indicates the results are significant at the .001 level.

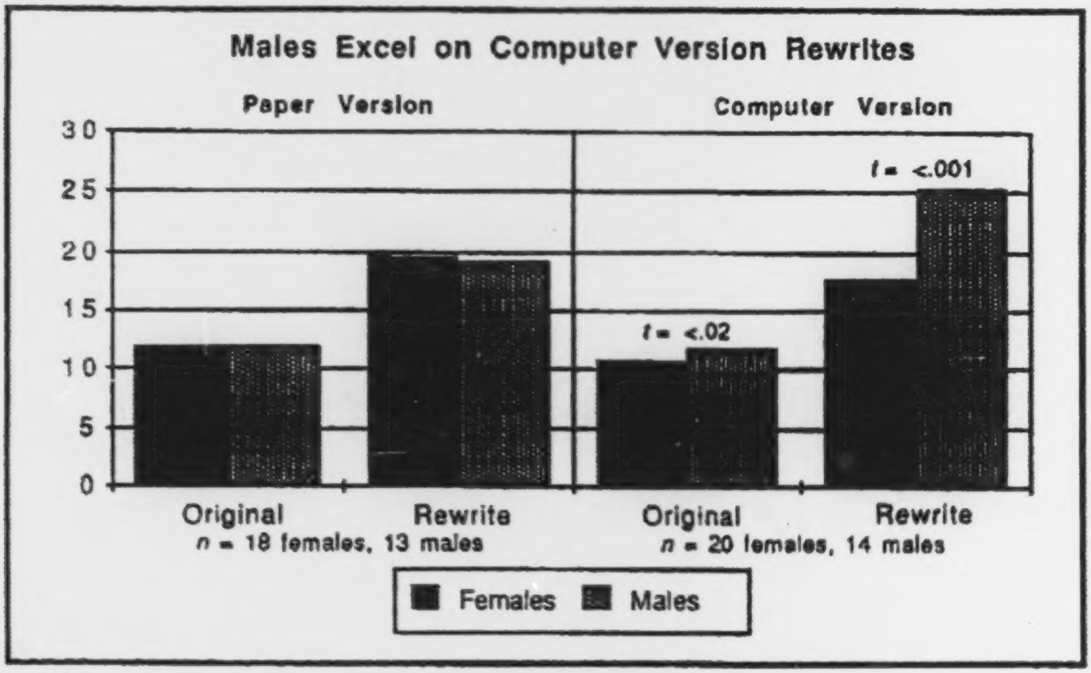


Chart 2.

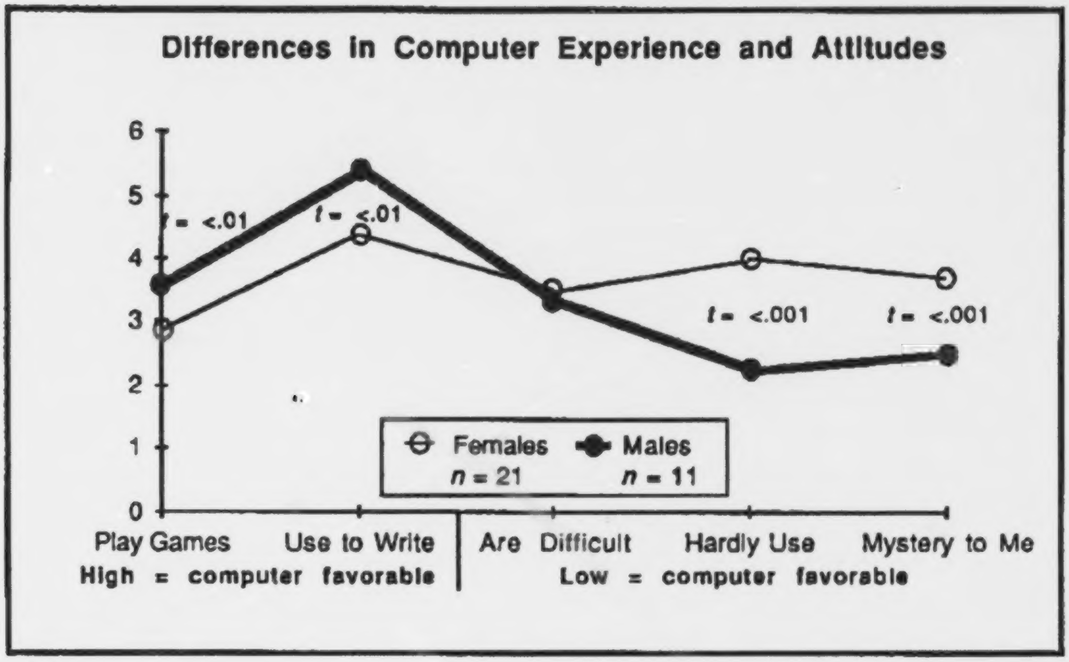


Chart 3.

In scoring that improvement, the computer-users also overcame a nearly one-point deficit ($t= <.01$) in the quality of their original leads compared to those of the paper group.

The results unexpectedly showed a strong difference in performance between male and female students on the computer version, but not on the paper version of the exercise. The average final score for the 14 computer-using males was 25.5, while for the 20 computer-using females it was 17.8 (t-test significant at .001 level). By contrast on the paper version the 18 females averaged a not-statistically-significant 4-tenths of a point better than the 13 males.

Females gain as many points rewriting on computer as they do rewriting on paper, but they start with lower original scores than males in the computer version ($t= <.02$).

The discrepancy in male and female scores was noted too late to pursue the question with the first students included in the study, but the four classes who participated most recently were surveyed after they completed the exercise to discern their experience with and attitudes toward computers.

Each student was asked to respond to the following five statements on a seven-point scale from *Strongly Disagree* to *Strongly Agree*:

1. I have frequently played computer games.
2. I have often used computers for writing or other tasks.
3. Computers generally are difficult to use.
4. I have hardly any experience with computers.
5. Computers are a mystery to me.

Uniformly, where a high score would indicate computer-familiarity, males scored higher than females—and where a low score indicated a greater level of comfort with the machines males scored lower than females. The differences were significant, except for the "difficult to use" item.

Discussion

The results of this study offer encouragement for further development of computer programs as tools for teaching writing. They suggest that it may not always be necessary to add to the instructor's hours of grading to provide additional meaningful writing instruction to students. Results of concurrent research, to be reported separately, suggest that students find such programs to be fun to use. They also report learning more from the program than from conventional classroom exercises.

The relatively low scores by women are troubling, but it would appear from the survey data that additional experience with computers may resolve that problem.[13]

---

[13] Conflicting research exists on this point. For a review of the literature and a report on a longitudinal study that found sex-based differences persist, see: Kathy A. Krendl, et.al., "Children and Computers: Do Sex-Related Differences Persist?" *Journal of Communication*, 39(3):85-93 (1989).

Many students in this study had absolutely no prior experience with Macintosh computers, and several noted on their survey forms that, while they had difficulty with the computer at the start of the test, they were finding it easy to use by the end of it.

The lower initial scores of the computer users may be a result of confusion about keyboarding techniques among some of the novice computer users. A few students initially put a return at the end of each line of their lead as if they were using a typewriter instead of a computer. That caused the computer to only score a small portion of what they had meant to be their lead, and substantially depressed their scores. It is also possible that eagerness to try out the computer feedback feature may have led some students to "jump the gun" a bit—not polishing their first version leads as much as they would have done had they been working on a typewriter.

It would appear computer-version scores could be further improved by enhancing the feedback the program provides. The computer's feedback, while more closely tailored to the student's copy, is less detailed than that of the paper version, and some comments from students in the study suggest they were sometimes left confused by the brief comments.

Further development is also needed to move toward a program that can be quickly adapted to work with different story content. All of the feedback code was custom written for the fact situation of Bayshore Blast, but many news leads have similar enough structures that once the key elements are identified, along with likely errors, it should be possible to "plug in" many different story scenarios.

Bibliography

Anderson, John R., C. Franklin Boyle and Brian J. Reiser, "Intelligent Tutoring Systems," *Science*, 228:456-462. (April 26, 1985).

Kearsley, Greg, ed., *Artificial Intelligence and Instruction: Applications and Methods*, (Reading, MA: Addison-Wesley, 1987).

Phillips, Gerald M. and Bradley R. Erlwein, "Composition on the Computer: Simple Systems and Artificial Intelligence," *Communication Quarterly*, 36(4):243-261 (1988).

Selfe, Cynthia L. and Billie J. Wahlstrom, "Computers and Writing: Casting a Broader Net with Theory and Research," *Computers and the Humanities*, 22:57-66 (1988).

Sleeman, D. and J. S. Brown, ed., *Intelligent Tutoring Systems*, (New York: Academic Press, 1982).

Integration of Product Development Cycle and Component Display Theory to Create Computer Aided Instruction

Robert Perkins
Assistant Professor of Education
School of Education, College of Charleston
Charleston, SC 29424
(803) 792-4840

Abstract

This project addresses the need for more educationally sound Computer Assisted Instruction (CAI) software. By using a systems approach to computer program development and combining it with instructional design theory, the educational value of CAI can be improved.

The Product Development Cycle developed by W. James Popham and Eva L. Baker (1971) is a systems approach to instructional product development. Popham and Baker established seven stages (Formulation, Item Specification, Item Tryout, Product Development, Product Tryout, Product Revision, and Operations Analysis) and rules that are to be followed at each stage. Formative and summative evaluation are incorporated to improve the product's development.

Instructional design is integrated into this process by using Component Display Theory developed by M. David Merrill. Merrill uses a Performance-Content matrix to develop learner objectives and items to test performance. Presentation Forms (Primary and Secondary) are used to develop the way information will be presented to students.

This paper reports how the Product Development Cycle and Component Display Theory were integrated to create an interactive videodisc CAI lesson to teach Introduction to Economics. The stages of development of the program and how the two theories were integrated in this process are described along with the timeline in which the events took place.

Introduction

In creating computer assisted instructional programs, it is important to take into account two processes. There must be an overall plan that establishes the developmental stages of the program from its beginning to the final version that will be used in the classroom. Learning theory must also be included in the developmental process at the appropriate stages. A process for developing the program and incorporation of instructional design theory are both necessary to increase the probability of a program being educationally sound. The development process involves following stages of development to ensure an organized process. Instructional design uses theory to present material to students in a way that has proven to increase learning. This project is based on the integration of a systems approach and instructional design theory in CAI development.

Product Development Cycle

Most systems approaches to program development processes are derived from the Basic Cybernetic Model (Logan, 1982). In this model, *Input* leads to *Processing* which in turn produces *Output*. This *Output* generates *Feedback* to be used by *Input* or *Processing* for any necessary alteration of the program. This is a continuous cycle, thus improving the quality of the system. One of these systems approach models is the Product Development Cycle proposed by Popham and Baker (1971). It specifies the steps to be followed through the developmental process and rules are also stated that should be followed at each of those steps. The stages in this model are as follows:

Stage I is *Formulation* which states what the program will accomplish and verifies a need for the program. The existence of other similar programs and their quality is researched at this stage (Popham & Baker, 1971).

Stage II involves the *Instructional Specification* of objectives. Stating what will be accomplished by the program in terms of learner objectives; learner entry, en-route and exit behaviors; and, criteria for student assessment are all part of this stage (Popham & Baker, 1971).

Stage III is *Item Tryout*. Prototype learner objectives, pretests, and posttests are administered to a group similar to the target population of the program using a lecture method of presentation. The strengths and weaknesses of the objectives and test items are assessed along with student attitudes toward the material (Popham & Baker, 1971).

Stage IV is the *Product Development* stage when the program is actually created. Allowing for learner practice, developing learner interest and motivation, and providing feedback are all taken into account during the developmental process. The concerns of teachers who may use the program must also be considered. Teacher attitude, classroom logistics and ease of use can all affect how the program will be received by the target audience (Popham & Baker, 1971).

Stage V is called *Product Tryout*. The program is tried in a classroom situation similar to the target group that the program is intended for. The strengths and weaknesses of the program are determined. The data is to be collected by the classroom teacher and interpreted by the program developer (Popham & Baker, 1971).

Stage VI is the *Product Revision*. This is when information collected at the Product Tryout stage is interpreted and necessary changes made to the program. Changes may involve rewording presentation of information, addressing learner objectives differently, or adjustments to test items to better reflect the learner objectives (Popham & Baker, 1971).

Stage VII is the last stage and it is called *Operations Analysis*. Summative evaluation is done to determine how well the original goals of the program have been met. Qualitative information from the students, the teachers involved as well as the experts in the field that the program addresses all contribute to this process (Popham & Baker, 1971).

Component Display Theory

Instructional design is "the process of deciding which methods of instruction are best for bringing about desired changes in student knowledge and skills for a specific student population" (Reigeluth, 1983, p. 7). Instructional design addresses instructional theory to accomplish learning in the most effective and efficient manner. There are many instructional design theories that have been proposed that are based on different learning theories. The instructional design theory used in this program was Component Display Theory (CDT) developed by M. David Merrill (1983, 1988).

Reigeluth (1983) describes Component Display Theory as a prescriptive process that specifies conditions, methods, and outcomes of instruction. "One of the outstanding features of CDT (aside from its highly integrative nature) is its basis for prescribing which model to use when" (Reigeluth, p. 281) in developing objectives.

Merrill (1983), explaining why CDT was developed, states:

> "Most descriptions of instruction identify three primary components: objectives, learning activities, and tests. Objectives state what is to be learned; learning activities are events in which the student must participate in order to acquire the objectives; and, tests are events that assess the degree to which the student acquired the objectives. The Component Display Theory is a set of prescriptive relationships that can be used to guide design and development of learning activities [which] promote acquisition of the objectives" (Merrill, p. 283).

Component Display Theory has two major goals. The first goal is to determine the learner objectives and how accomplishment of the objectives will be assessed. The second goal is to specify the learner activities that will be used to accomplish the objectives. By using Component Display Theory, the developer is guided through these two goals in an organized prescriptive manner.

Merrill (1983) uses a two-dimensional matrix to prescribe the learner objectives (see Figure 1). This matrix is formed by: (a) the type of *Content* (Facts, Concepts, Principles, and Procedures); and, (b) the desired level of student *Performance* for that particular objective (Remember, Use, and Find).

In the performance dimension of the matrix, the different levels specify different types of student response. *Remember* requires that students be able to recall exact factual information. Applying the information to a specific situation is what the student is required to do at the *Use* level. The *Find* level requires students to synthesize a completely new situation using their knowledge (Merrill, 1983).

Content forms the second dimension of this matrix. The first level of content is *Fact*, or factual knowledge. The second level of content is *Concept*. These are groups of things that share similar characteristics. The next level is *Procedure*, which are the steps that

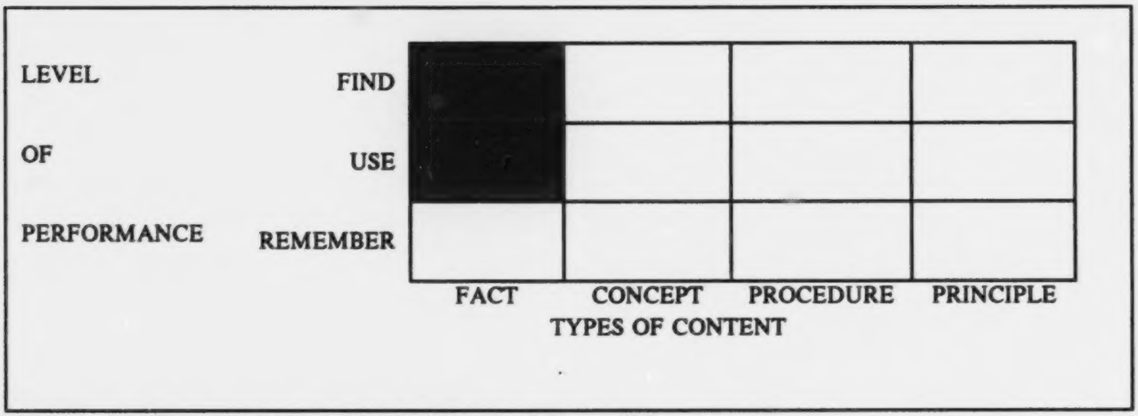


Figure 1. Performance-Content Matrix (Merrill, 1983, p. 286)

accomplish a given task. The last level of content is *Principle* which are what establishes relationships between concepts. They can be interpreted to make predictions (Merrill, 1983).

Specifying Objectives. Learner objectives are generated from the Performance-Content matrix. There is no abstract or general representational quality to facts, therefore there is no Use-Fact or Find-Fact level in the matrix and no learner objectives are specified for these two parts of the matrix (Merrill, 1983).

Merrill (1983) gives, as examples, the following matrix cells and a corresponding activity for students: *Remember-Fact* (On a topographic map, what is the symbol for a church?); *Remember-Concept* (What are the characteristics of a conifer?); *Use-Concept* (Is the mountain pictured in this photograph an example of a folded mountain?); *Find-Concept* (Sort the rocks on this table into several different piles. Indicate the characteristics by which one of your classmates could sort them into the same piles.); *Remember-Procedure* (What are the steps in balancing a checkbook?); *Use-Procedure* (Demonstrate how to clean a clarinet.); *Find-Procedure* (Write a computer program that will index and retrieve recipes.); *Remember-Principle* (Explain each of the three projection techniques for making maps on the earth's surface.); *Use-Principle* (Read the following case study of an ecological system. In this system the rodents are increasing in number. Predict some possible hypotheses based on your knowledge of life cycles and the interdependence of species in this ecological system.); and, *Find-Principle* (Set up an experiment to assess the effect of tobacco smoke on plant growth. Report your findings.) (p. 288-289).

Specifying Test Items. Merrill (1983) states that while the learner objectives are being created from the Performance/Content matrix, the test items should be developed at the same time since the test items are to test whether the learner objective was met. "An objective, when adequately specified, defines a class of acceptable test items, each of which should provide some degree of measurement for the performance-content combination represented by the objective" (Merrill, p. 291). Important at this stage of development of the program is the number of test items to be used and the criteria for successfully accomplishing the objectives.

The number of test items to measure accomplishment of learner objective is determined by the objective itself. When students are asked to recall factual information or if something is to be remembered word for word, only one test item per objective is required. Use level and Find level objectives will require more test items to assess understanding. These items should be varied reflecting different situations and difficulty levels.

Criterion to show successful acquiring of the objectives should vary depending on what and how material is to be learned. Factual information or information that is to be recalled word for word should have a 100% criterion level. For answers that are paraphrased, lower criterion should be acceptable to reflect the students interpretation of the question and the material. Merrill (1983) suggests a split criterion for the upper level test items to reflect different difficulty levels.

Component Display Theory also prescribes the way that the new material will be presented to students. Primary and Secondary Presentation Forms are used to develop the way material will be delivered to students. Gagné and Briggs (1979) state that different types of learner objectives require different strategies for the objectives to be learned. Merrill (1983) takes their theory a step further by creating the Presentation Forms that prescribe to the developer what the different strategies will be. Merrill states "the task for the designer is not to invent an objective but rather to select that objective that corresponds to the intended performance-content level" (p. 290).

Primary Presentation Forms. Merrill (1983) uses a two-dimensional matrix to determine how material will be presented to students depending on: (a) specificity of the subject matter (generality or instance) and (b) how the student will respond (expository or inquisitory) (see Figure 2). *Generalities* (rule) are statements of definition, the steps in a procedure, and principles. *Instances* (examples) may include a specific object, symbol, process, procedure or event that is determined by a concept class that can be used to identify that concept (Merrill, 1983, 1988). *Expository* presentations refer to a situation where a student is told, illustrated, or showed the presentation. *Inquisitory* is when a student is expected to complete a statement or apply a general case to determine a specific situation (Merrill, 1983).

Secondary Presentation Forms. "Primary Presentation Forms are the major vehicle of the instruction. Secondary Presentation Forms are those methods that are used to facilitate the students' processing of the information or to provide items of interest, such as contextual background" (Merrill, 1983, p. 308). Secondary Presentation Forms provide more detail, prerequisite information, feedback, or help as necessary to students to aid in the learning process. The Secondary Presentation Forms prescribe how and when that additional information is to be delivered (see Figure 3).

Integration of Product Development Cycle and Component Display Theory

Integrating the Product Development Cycle with the Component Display Theory produced an over-all plan which incorporated instructional design procedures in CAI development. The rules associated with the Product Development Cycle specified what the responsibilities of those involved in the development were at each

	Tell or Expository (E)	Question or Inquisitory (I)
Generality (G)	Rule or Generality EG TG	G-Practice IG G-Test QG
Instance (eg)	Example Eeg Teg EG	eg-Practice Ieg eg-Test Qeg

Figure 2. Primary Presentation Forms (Merrill, 1983, p. 306)

stage. Component Display Theory specified how the objectives were to be developed and how the material was to be presented to students.

The results of integrating the two processes produced the following steps:

Stage I Formulation

The topic of the instructional program was determined (Introduction to Economics). Information from experts in the high school where it would be piloted and university level content experts were interviewed to determine the need for this type of program and to develop a survey to determine objectives. This survey was sent to other experts (an economics professor, a banker, and other university social studies professors) to get a greater consensus for the goals of the program.

The largest source of information in economics was *A Framework for Teaching the Basic Concepts: Master Curriculum Guide in Economics* which was developed for the Joint Council on Economic Education (Saunders, Bach, Calderwood, & Hansen, 1984). This source was suggested by two of the university experts. The videodisc for the lesson was also selected, *Introduction to Economics* by MECC.

This lesson was to use interactive videodisc to present the lesson by "repurposing" a commercially available videodisc. Determining a lesson that (a) there was a need for and (b) had an available videodisc took

	EG	Eeg	Ieg	IG
CONTEXT (c)	EG'c	Eeg'c	Ieg'c	IG'c
PREREQUISITE (p)	EG'p	Eeg'p		
MNEMONIC (mn)	EG'mn	Eeg'mn		
MATHEMAGENIC HELP (h)	EG'h	Eeg'h	Ieg'h	IG'h
REPRESENTATION (r)	EG'r	Eeg'r	Ieg'r	IG'r
FEEDBACK (ca) correct answer (h) help (u) use			FB'ca FB'h FB'u	FB'ca FB'h FB'u

Figure 3. Secondary Presentation Forms (Merrill, 1983, p. 307)

a long time, from September 15,1988 to December 23, 1988 when the disc finally arrived. Developing the survey and compiling the results to determine the objectives took from December 23, 1988 to January 22,1989.

Stage II Instructional Specifications

Using the Performance/Content matrix of Component Display Theory and the surveys from experts in the field, learner objectives, and items to test accomplishment of the objectives were developed. Using Primary and Secondary Presentation Forms, the way that material was to be delivered to students was developed.

Developing the learner objectives took from January 23, 1989 to February 3, 1989. The initial test items were selected from an economics textbook from January 27, 1989 to February 3, 1989. The way that the information would be presented to students using CDT was developed from February 3, 1989 to February 15, 1989.

Stage III Item Tryout

The students in a pilot class took a pretest, received instruction using a lecture method, and were administered a posttest to determine the quality of the objectives and test items. Students' attitude toward the instruction and subject matter and teacher attitude information was collected to be used in further development of the program.

The pretest was administered on February 15, 1989 and the lesson was taught from February 16 to 17, 1989. The posttest was given on February 20, 1989. The first version of the test was developed using test items from a textbook. These items proved not to match the objectives that this lesson was teaching. New test items were created by reviewing the objectives again from February 15, 1989 to March 1, 1989. The classroom teacher was interviewed for suggestions to improve the lesson.

Stage IV Product Development

Creating the computer program was done at this stage. After reviewing information collected at the Item Tryout stage, some objectives and test items were modified. Component Display Theory and the Primary and Secondary Presentation Forms again played an important part in creating the program by adapting the objectives and presentations to computer delivery of the instruction.

The actual programming process took from February 22, 1989 to March 23, 1989. Knowing how to create text programs in Pascal was the limit of my experience until this project. Programming the Apple IIGS desktop environment caused many problems that had to be overcome. Creating the drivers for the Pioneer LD-V4200 Laserdisc player also had to be accomplished.

Stage V Product Tryout

A prototype version of the computer program was presented to the students in the class that was lectured to in the pilot study. Information collected from students included responses, discussions about how they arrived at their decisions, reaction to the method of presentation, and attitudes toward the program. Information collected from the teacher included attitude toward the program, the method of presentation, and classroom logistical data (imposition on the classroom situation, amount of teacher involvement, and difficulty in operating the equipment).

The first test of the program was a failure. The Pascal programming environment had initialized the videodisc player for me during the development process. When the program was first tried, the initialization process did not take place and the laserdisc player did not respond to commands. This test was on March 23, 1989.

After the reason for the failure of the program was determined, the laserdisc driver was rewritten and a new, successful test of the program took place on April 3 and 4, 1989 and the posttest was administered on April 5, 1989.

Stage VI Product Revision

The information that was collected at the previous stage was interpreted and incorporated into the revised version of the program. Again, Component Display Theory played an important part in adjusting objectives, test items, and presentation of material. Matching of objectives and test items as well as adequate presentation of material were taken into account and necessary adjustments were made.

The program was revised from April 3 through April 6, 1989. Two of the screens of information needed to be reworded to be better understood by the students. Spelling errors were also corrected. Since this program relied on a correct answer pool, more answers were added from student responses.

Stage VII Operations Analysis

The completed program was administered to students in a different class. Student pretest and posttest data were collected along with attitudinal information from teachers and students. Qualitative information from the experts at the high school as well as university level experts all contributed to the summative evaluation.

The pretest for the last stage of the development process took place on April 4, 1989. The final presentation of the program took place on Wednesday April 5 and Friday, April 7, 1989. The attitude survey was administered on April 7, 1989 and the post test was administered on April 10, 1989. A test for retention was administered on April 21, 1989.

Conclusions

Some of the problems that I had, especially those involving programming, are no longer that serious because of the development of authoring languages such as HyperCard, Hyper Studio, Tutor Tech, and Linkway. Many of the things that I had to program (the question screens, response windows, and the laserdisc drivers) are built into the authoring language and are easily accessible to the programmer.

The integration process that I used does not readily lend itself to the repurposing of laserdiscs or even to the development of daily lessons by classroom teachers. The process is too lengthy for that. It is, however, viable for the development of commercial software. Because of the formative and summative evaluation processes and the step by step procedures, the end product is likely to be of better quality.

The one primary thing emphasized to me during this entire process was the need for a team approach to software development. A programmer, instructional designer, and content area expert all need to be part of the team. This would accomplish two things. First, none of these areas would be slighted to make it easier for another. The objectives would not have to be adjusted because they would be difficult to program, for example. Second, with a team approach, the process would not be as long. Many of the steps could be going on concurrently.

By using a process that includes a systems approach which incorporates instructional design theory to create computer assisted instruction programs, educationally sound computer assisted instruction is more likely to be developed. Both the systems approach and the instructional design process brought important considerations to the design of this CAI program.

The systems approach to program development included both formative and summative evaluation procedures that provided much information during program development. The step by step nature of this approach forces the designer to be methodical in the development process and to carry out evaluations at the appropriate stages.

A systems approach alone is not enough to guarantee the success of a program because of its generic nature. It creates an overall plan of attack, but does not deal necessarily with learning theory. A trial and error approach toward learning develops (Output leading to Feedback into Input) that may eventually lead to educationally sound programs but does not guarantee it. By including learning theory from the beginning, the program is more likely to be educationally sound from the beginning. With the body of knowledge developed from research on learning objectives and learning strategies, it makes sense to incorporate that knowledge into the development of CAI.

By combining these two processes, two things contribute to better CAI programs. A system approach to this process brings both summative and formative evaluation procedures. Instructional design brings those learning strategies that best contribute to education. Those parts of the program that need correcting are done while the program is being created, not after it has been marketed.

References

Baker, E. L. and Schutz, R. E. (Ed.). (1971). *Instructional Product Development*. N.Y., N.Y.: Van Nostrand Reinhold Co.

Gagné, R. M. & Briggs, L. J. (1979). *Principles of Instructional Design* (2nd ed.). New York: Holt, Rinehart and Winston.

Jonassen, D. H. (Ed.). (1988). *Instructional Designs for Microcomputer Courseware*. Hillsdale, N.J.: Lawrence Erlbaum Associates, Publishers.

Logan, R. S. (1982). *Instructional Systems Development*, N. Y., N. Y.: Academic Press.

Merrill, M. D. (1988), Applying Component Display Theory to the design of courseware in Jonassen, D. H. (Ed.). *Instructional Designs for Microcomputer Courseware*. Hillsdale, N.J.: Lawrence Erlbaum Associates, Publishers, 61-95.

Merrill, M. D. (1983). Component Display Theory in Reigeluth, C. M., (Ed.). (1983). *Instructional Design Theories and Models: An overview of their Current Status*, Hillsdale, N.J.: Lawrence Erlbaum Associates, Inc., 279-334.

Popham, W. J., and Baker, E. L. (1971). Rules for the development of instructional products in Baker, E. L. and Schutz, R. E. (Ed.). *Instructional Product Development*. N.Y., N.Y.: Van Nostrand Reinhold Co., 129-168.

Reigeluth, C. M., (Ed.). (1983). *Instructional Design Theories and Models: An overview of their Current Status*, Hillsdale, N.J.: Lawrence Erlbaum Associates, Inc.

Reigeluth, C. M. (1983). Instructional design: What is it and why is it? in Reigeluth, C. M. (Ed). *Instructional Design Theories and Models: An overview of their Current Status*, Hillsdale, N. J.: Lawrence Erlbaum Associates, Inc., 3-36.

Saunders, P., Bach, G. L., Calderwood, J. D., & Hansen, W. L. (1984). *A Framework for Teaching the Basic Concepts: Master Curriculum Guide in Economics*, N.Y., N.Y.: Joint Council on Economic Education.

Teaching and Learning with Computer Based Instruction: Findings from New York City's Computer Pilot Program

Karen Swan
State University of New York at Albany

Marco Mitrani
Teachers College, Columbia University

Frank Guererro and John Schoener
Office of Research, Evaluation and Assessment
New York City Board of Education

Abstract

The Computer Pilot Program is an on-going project designed to explore the use of comprehensive computer based instructional systems (CBI) for the remediation of basic skills deficiencies among educationally disadvantaged students in New York City's public school system. To date, interviews have been conducted with 197 teachers and 718 students participating in the program and the standardized test scores of 3,795 students have been included in analyses of 14 different systems placed in 12 elementary, 8 junior high/intermediate, and 12 high schools located throughout New York City. The results of these analyses support the efficacy of CBI for the delivery of basic skills remediation to educationally disadvantaged student populations. They indicate that (1) involvement with CBI programs can result in significant and meaningful increases in the academic performance of educationally disadvantaged students; that (2) CBI can be equally effective in increasing both their reading and mathematics performances; that (3) among such population, an inverse relationship exists between students' instructional level and their achievement gains resulting from CBI use; but that (4) at a high school level, process-oriented programs may be more effective than the more common drill and practice CBI. Our findings also suggest that the use of comprehensive CBI is indeed altering the ways in which teachers teach and students learn. We found that the computer based classroom environments we visited were more student-centered and cooperative, that teachers were more the facilitators of learning and that learning was more individualized when done using computers, and that students were more motivated and less threatened when learning on computers than when learning in regular classroom settings. It is our belief that the success of the CBI programs we evaluated derives at least in part from the effects of such environments on the educationally disadvantaged students involved in them.

Background

In an educational system as large and diverse as New York City, traditional methods of education may not succeed with all students. In particular, educationally disadvantaged students in need of basic skills remediation might benefit from alternative means of instruction. One such alternative involves computer based instruction (CBI). Indeed, research suggests that comprehensive CBI might best be used for delivering basic skills remediation to educationally disadvantaged students (Jamison, Suppers & Wells, 1970; Chambers & Sprecher, 1980; Schmidt, Weinstein, Niemiec & Walberg, 1987; Swan, Mitrani, Guerrero & Schoener, 1989). Indications are also that students learn more quickly from CBI programs than from regular classroom instruction (Ragosta, 1982; Kulik, Kulik & Bangert-Drowns, 1985; Bangert-Drowns, Kulik & Kulik, 1985), and that their motivation for and attention to needed drill and practice work is improved by the use of computer delivery (Electronic Learning Laboratory, 1982; White, 1986; Niemiec & Walberg, 1987). The Computer Pilot Program was designed to investigate such educational use of computers with New York City's educationally disadvantaged student population.

The Computer Pilot Program was funded by the Division of Computer Information Services of the New York City Board of Education in conjunction with the vendors of 14 computer based instructional programs—Autoskills, CCC, CCP, CNS, Degem, ESC, Ideal, New Century, PALS, PC Class, Plato, Prescription Learning, Wasatch, and WICAT. It is an on-going program targeted for an educationally disadvantaged student population requiring supplemental basic skills instruction in grades three through twelve. The goals of the Computer Pilot Program are to identify comprehensive computer based instructional programs which can be effective in increasing the academic performance, attendance, and positive attitudes of educationally disadvantaged and/or at-risk students, as well as implementation factors significantly contributing to the success of such programs.

Vendors placed their systems in appropriate schools, and offered staff development, equipment maintenance, and support at little or no cost for the evaluation period. Individual schools were responsible for staffing the programs and providing time for staff development. Participating schools were also responsible for selecting a target group of students in need of remediation in basic reading and/or mathematical skills, and for scheduling that group in compliance with the stated needs of the particular system they were using. The Division of Computer Information Sciences provided additional equipment, technical assistance, and coordination among groups involved in the Computer Pilot Program. Evaluation of the Computer Pilot Program was carried out by the Office of Research, Evaluation, and Assessment/Instructional Support Evaluation Unit (OREA/ISEU) of the New York City Board of Education.

Methodology

Empirical analyses of student performance gains were made using comparisons of participating students' percentile scores on citywide tests given in 1987, 1988 and 1989. Spring reading and mathematics performance scores were compared for the year preceding and the year of students' participation in the Computer Pilot Program. Tests used were the Degrees of Reading Power (DRP), and the Metropolitan Achievement Test (MAT). The scores of 417 participating students were included in the analyses of reading achievement (1057, elementary; 631 intermediate; 459, high school) and the scores of 1639 participating students were included in the analyses of mathematics achievement (1068, elementary; 561, intermediate). Matched *t*tests were used to test for significant differences between students 1987 and 1988 or 1988 and 1989 reading and mathematics scores, and effect sizes for the mean differences between these generated. Effect sizes were calculated by dividing the mean differences by the standard deviations of the mean differences. Effect sizes of one or greater thus indicate performance increases of a full standard deviation or more.

In addition, we interviewed a sample of students and teachers at each school participating in the Computer Pilot Program in the fall and again in the spring of the 1987-88 and 1988-89 school years. Open-ended interviews were conducted with whole classes of students participating in the program to determine students' responses to it. Questions asked were concerned with what they liked the most and the least about the Computer Pilot Program, how they thought learning on computers was different from regular classroom learning, and whether or not they thought their involvement in the program was changing their school performance, their attitudes toward school and learning, and/or their school attendance. Participating teachers were interviewed individually to determine both their responses to the program and how they believed their students were responding. Students and teachers were also asked to individually complete a questionnaire designed to corroborate and quantify attitudinal information gained in the open-ended interviews. Five hundred and fifty-two students and 150 teachers completed this first questionnaire. Their responses to each question were tabulated and the percentage responding with each choice calculated.

A second set of questionnaires was given to students and teachers interviewed during the 1988-89 school year. These were designed to assess participant perceptions of the social context of the computer room. It asked eleven questions concerning how they believed learning on computers differed from regular classroom instruction. Ninety-nine students and 17 teachers completed this second questionnaire. Their responses to each question were tabulated and the percentage responding with each choice calculated.

Finally, observations of students and teachers using CBI were conducted in each participating school during the 1988-89 school year, and the same students were observed during regular classroom activities. Whenever possible, regular classroom observations also included the same teachers as observed in the computer room, but in some cases this was not possible because CBI teachers never left the computer room. All the regular classroom activities observed were none-the-less concerned with content similar to that addressed by the CBI programs evaluated (reading and mathematics). Observations lasted fifteen minutes, and consisted of the recording of all student-teacher interactions occurring during that period of time. These were characterized as either student-initiated or teacher-initiated, whole group or individual, and involved with either content or process questions. The total number of interactions in each category was tabulated and used to calculate ratios of teacher-initiated to student-initiated, whole group to individual, and content to process interactions for both CBI and regular classroom instruction. The total numbers of student-student interactions occurring in each environment were also recorded. The significance of differences in these sorts of interactions between computer and regular classrooms was assessed using a chi-square analysis.

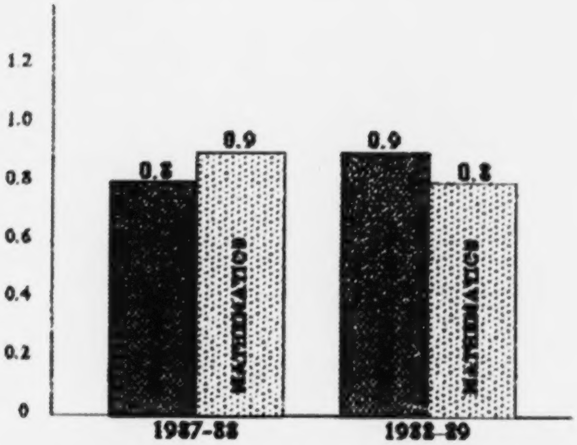


Figure 1. Overall Achievement Gains 1987-88 and 1988-89 Evaluations

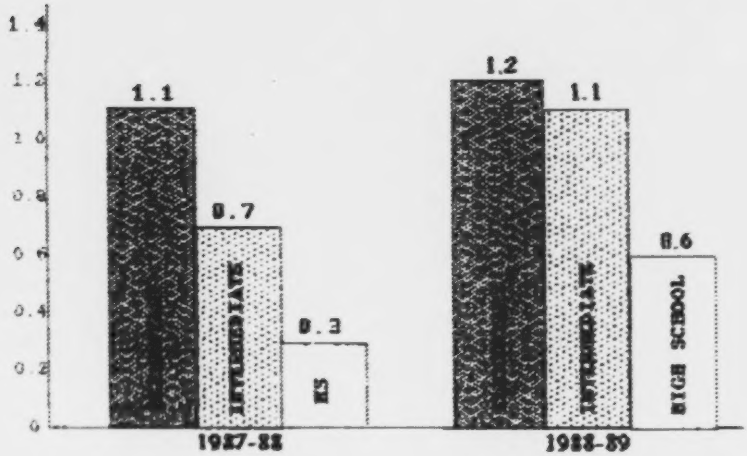


Figure 2. Reading Achievement Gains 1987-88 and 1988-89 Evaluations

Results

The average effect size for participating students' reading achievement gains in the 1987-88 school year was 0.8; the average effect size for participating students' reading achievement gains in the 1988-89 school year was 0.9. During the 1987-88 school year the average effect size for participating students' mathematics achievement gains was 0.9; in 1988-89, it was 0.8. Achievement gains thus averaged nearly a full standard deviation, indicating that the effects of the Computer Pilot Program were not only significant but meaningful. These are shown in Figure 1 which compares achievement gains from the 1987-88 evaluation with those of the 1988-89 evaluation. Notice that effects of CBI were about equal for reading and mathematics instruction. Notice also that although in 1987-88, mathematics gains showed slightly greater effects than reading gains, in 1988-89 reading gains exhibited slightly greater effects than mathematics gains, thus, achievement gains were relatively consistent overall. Such results reflect the utility of CBI for delivering instruction in both reading and mathematics, at least to the educationally disadvantaged population we tested.

Figures 2 and 3 compare average effect sizes for reading and mathematics increases between grade levels for both the 1987-88 and 1988-89 school years. Notice the similarities between these, and that in all of them, effect sizes decrease as instructional levels increase. The inverse relationship between instructional level and achievement increases resulting from CBI use first reported by Kulik (1981), thus quite clearly seems applicable within educationally disadvantaged populations. Such results indicate that, as is the case with general student populations, the most effective use of CBI for the basic skills remediation of educationally disadvantaged students is at the elementary school level, although among the 1988-89 data in particular, there is some indication of a threshold effect occurring among third graders. That is, it is possible that CBI may not be useful to students at or below the third grade level, most likely because of difficulties such students experience in operating CBI programs.

Notice also differences in effect sizes for the reading achievement of high school students between the 1987-88 and 1988-89 evaluations. While these were not meaningful and in most cases not significant in 1987-88, they were significant and at least approaching meaningfulness in 1988-89. These 1988-89 findings are based almost entirely on a single, process-oriented program, Wasatch. Although producing significant results among elementary and intermediate school students, Wasatch was one of the least effective programs tested at these levels. It was, however, the most effective program tested with high school students. It seems likely that Wasatch's success at the high school level (as well as its lesser effectiveness at lower instructional levels) derives from its process-orientation. While the more common learning-by-objectives programs are arguably more effective at lower instructional levels where basic skills can be adequately

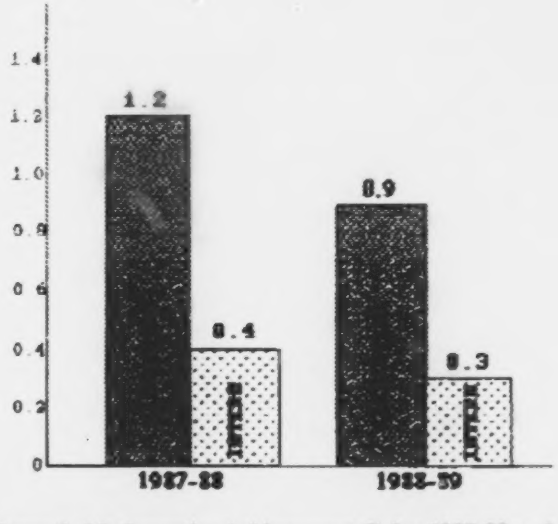


Figure 3. Mathematics Achievement Gains 1987-88 and 1988-89 Evaluations

Table 1

STAFF ATTITUDES SURVEY: OVERALL RESULTS*
N - 175

	not at all	very little	somewhat	quite a lot	totally
correlates w/ curriculum?	.03	.05	.20	.37	.35
integrated into classroom?	.03	.03	.29	.47	.18
feedback useful?	.06	.04	.17	.31	.42
training useful?	.14	.03	.23	.30	.30
usefulness to students?	.00	.04	.12	.35	.45
improvedattitudes?	.02	.03	.04	.31	.41
improvedattendance?	.07	.10	.32	.27	.28
improvedachievement?	.08	.03	.04	.30	.34

* Figures in Percentage of Total Staff Members Interviewed

Table 1.

addressed through the drill and practice type exercises typical of such programs, it is possible that the development of the more complex, higher order-skills required at higher instructional levels is not adequately addressed by such exercises. The holistic, multi-process exercises found in process-oriented programs such as Wasatch may be better suited for students at these levels.

Implementation factors that may have effected program success include: physical setting, overall program size, the number of computers employed, student/staff ratios, time spent on computers, the involvement of classroom teachers, and serious staffing problems. The most successful program implementations seemed to involve the location of

Table 2

STUDENT ATTITUDES SURVEY: OVERALL RESULTS*
N - 630

My involvement with this program changed my . . .

	YES	NO	MAYBE
ATTITUDE	.63	.28	.09
SCHOOLWORK	.76	.19	.05
ATTENDANCE	.53	.39	.08
SELF-IMAGE	.53	.40	.07

My involvement with this program . . .

	strongly disagree	disagree	no opinion	agree	strongly agree
keeps me in school	.08	.24	.26	.27	.15
makes me want to come to school	.09	.28	.22	.25	.16
makes me feel better about school	.06	.19	.22	.36	.17
changed my attitude	.07	.22	.29	.29	.13
helps me learn	.02	.05	.09	.45	.39
helps me get better grades	.05	.13	.26	.32	.24
helps me feel better about myself	.05	.14	.29	.32	.20

* Figures in Percentage of Total Students Interviewed

Table 2.

programs within larger educational settings (as opposed to in isolated computer labs), enrollments of between 50 and 100 students, a 15/1 student/staff ratio, and between 60 and 135 minutes per week scheduled for computer use. Programs facilitated by computer specialists tended to be more successful than those implemented by regular classroom teachers. Implementation factors that did not seem to effect program success include: serious equipment problems, years in operation, student/computer ratios, correlation with general curricula, integration into regular

Table 3

HOW IS COMPUTER-BASED LEARNING DIFFERENT FROM LEARNING IN A REGULAR CLASSROOM?

		MORE	LESS	SAME
INDIVIDUAL ATTENTION ?				
	TEACHERS	88%	0%	12%
	STUDENTS	37%	14%	49%
DISCIPLINE ?				
	TEACHERS	0%	100%	0%
	STUDENTS	7%	69%	24%
UNDERSTANDING OF MATERIAL?				
	TEACHERS	82%	0%	18%
	STUDENTS	57%	7%	36%
INTEREST ?				
	TEACHERS	94%	0%	6%
	STUDENTS	76%	4%	20%
LEARNING TIME ?				
	TEACHERS	0%	71%	29%
	STUDENTS	20%	41%	39%
COOPERATIVE LEARNING ?				
	TEACHERS	59%	18%	23%
	STUDENTS	28%	46%	26%
THREAT ?				
	TEACHERS	0%	71%	29%
	STUDENTS	11%	67%	22%
EMBARRASSMENT ?				
	STUDENTS	5%	81%	22%
CONTROL OVER LEARNING ?				
	STUDENTS	63%	13%	24%
NOISE ?				
	STUDENTS	8%	77%	15%
AWARENESS OF INDIV. PERFORMANCE?				
	TEACHERS	82%	0%	18%
CLASSROOM MANAGEMENT ?				
	TEACHERS	29%	65%	6%
COMFORT ?				
	TEACHERS	29%	12%	59%
COMPUTER-BASED LEARNING?				
	TEACHERS	82%	6%	12%
	STUDENTS	68%	11%	21%

Table 3.

classroom activities, utilization of program feedback, and perceived usefulness of training.

Overall staff attitudes toward CBI are summarized in Table 1. The results are very positive, particularly concerning the usefulness of program involvement to participating students. The staff members interviewed believed that CBI was most useful in improving students' attitudes and least useful in improving their attendance. A majority believed that CBI involvement was resulting in improved academic achievement among the students they taught. The staff members interviewed also thought that the content covered by the CBI programs was well correlated with regular school curricula, hence was being well integrated into regular classroom activities. Over forty percent believed that the diagnostic feedback provided by comprehensive CBI was extremely useful, but a similar number also indicated that more training was needed before they could take full advantage of such systems.

Table 2 summarizes overall student attitudes toward CBI. The students interviewed overwhelmingly agreed with the statement, "My involvement with this program helps me learn." Such finding corroborates the open-ended interviews in which students frequently told us that what they liked best about CBI was that it helped them learn. Indeed, in contrast to their teachers, the students interviewed believed that their involvement with CBI was most useful in improving their schoolwork. Students agreed with their teachers that it was least useful in improving their attendance.

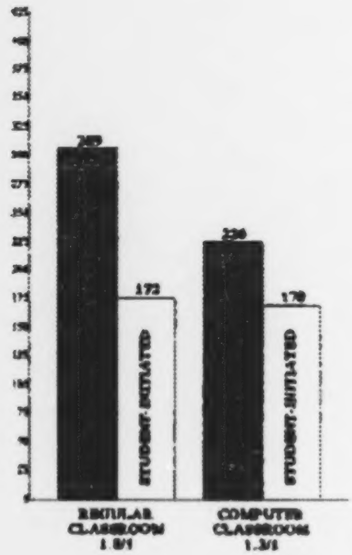


Figure 4. Ratio of Teacher-Initiated to Student-Initiated Interaction in Regular and Computer Classrooms

Table 3 summarizes teacher and student responses to the questionnaire concerned with the social context of computer based learning. It shows that teachers and students agreed that students were less threatened by, more interested in, and understood better material presented on computers, and that discipline was less of a problem when learning was computer based. Teachers also believed that they gave students more individual attention, were more aware of individual performance, and that classroom management was less of a problem in computer classrooms. They also thought that students took less time to learn when learning from computers than when learning in regular classrooms, and that more cooperative learning took place in computer based classrooms. Students thought they were more in control of their own learning and less embarrassed by computer based learning, and that computer classrooms were less noisy than regular classrooms. Both teachers and students agreed that more learning should be done on computers.

The chi-square analysis of student-teacher interactions revealed significant differences between regular and computer based classrooms in the numbers of student-initiated interactions (X^2 [1,5] = 56, $p < .001$), individual interactions (X^2 [1,5] = 375, $p < .001$), and interactions concerned with processes (X^2 [1,5] = 28.1 $p < .001$). The quality of these differences is indicated by the ratios of the various kinds of interactions found in differing classroom environments.

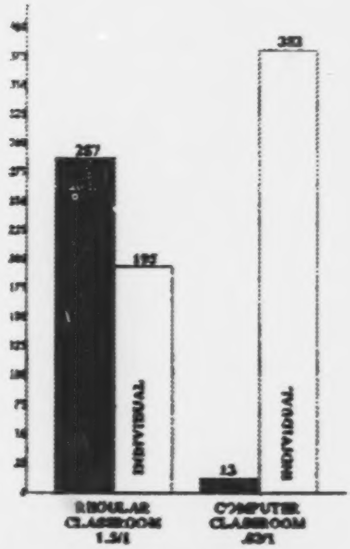


Figure 5. Ratio of Group to Individual Interactions in Regular and Computer Classrooms

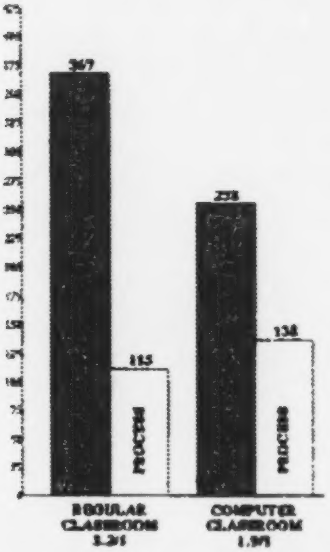


Figure 6. Ratio of Content to Process Discussions in Regular and Computer Classrooms

Figure 4 shows the ratios of teacher-initiated to student-initiated interactions occurring in regular and computer based classrooms. The ratio of teacher- to student-initiation was 1.8/1 in regular classrooms and 1.3/1 in computer rooms. The results indicate that although the numbers of student-initiated interactions were almost identical during regular and computer based instruction, there were many more teacher-initiated interactions in regular classrooms. It should be noted that these findings do not indicate more learning or even more directed learning taking place in regular classrooms, because in computer based learning, the computer is a teacher. It is also important to note that these results show that neither are human teachers superfluous in such environments. Because students felt they had more control over computers than they did over human teachers, however, such results corroborate student perceptions of greater control over their own learning when learning on computers.

Figure 5 shows the ratios of group to individual interactions occurring in these differing classroom environments. The ratio of whole group to individual interactions was 1.5/1 in regular classrooms and .03/1 in computer rooms. These results reveal that whereas the majority of interactions occurring during regular classroom instruction were whole group, the overwhelming majority of interactions occurring during CBI were individual. They thus corroborate teachers' beliefs that they gave students more individual attention and were more aware of individual students' performance in computer environments than in regular classroom settings. It should be noted that although most of the students we interviewed (49%) thought that they received about the same amount of individual attention from teachers when learning on computers, many more (38%) thought they received more such attention than thought they received less (13%). It may be that the far greater amounts of individual attention they actually received during computer based learning, together with the greater degree to which they initiated such interaction, contributed more to students' sense of greater control over their own learning than it did to their sense of individual attention received.

The ratio of interactions concerned with content to those concerned with process was 3.2/1 in regular classrooms and 1.9/1 in computer classrooms. Figure 6 illustrates these. The findings reveal that although the numbers of process-related interactions were about the same in the two environments, there were far more content-related interactions in regular classrooms than there were in computer classrooms. It should be noted, however, that these results do not indicate that less time was devoted to content learning in computer classrooms because learning from computers probably makes up the difference in the amount of content-related learning taking place in computer based classrooms. Because one might expect far more process-related interactions among students involved with a new and technically demanding medium like computers than in the regular classrooms to which they are accustomed, it is also perhaps surprising that there was so little disparity between process-related interactions occurring in the two environments. The results, then, indicate that computers are being accepted by students and integrated into regular school activities quite easily.

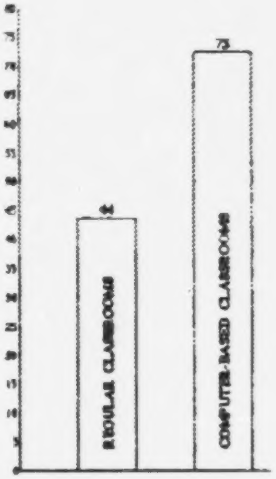


Figure 7. Student-Student Interactions in Regular and Computer Classrooms

The total number of student-student interactions observed in regular classrooms was 44. The total number of student-student interactions observed in computer rooms was 73. Figure 7 shows these. These results indicate that, even though comprehensive CBI is clearly not a form of computer based learning designed to encourage cooperative learning, more cooperative learning was taking place in the computer rooms we observed. Such findings support teachers' beliefs that their students shared more learning when learning on computers. In light of these findings, it is curious that the students we interviewed did not share their teachers' views.

Discussion

The results of the empirical analyses support the efficacy of CBI for the delivery of basic skills remediation to educationally disadvantaged student populations. They indicate that involvement with CBI programs can result in significant and meaningful increases in the academic performance of educationally disadvantaged students; that CBI can be equally effective in increasing both their reading and mathematics performances; that among such population, an inverse relationship exists between students' instructional level and their achievement gains resulting from CBI use; but that, at a high school level, process-oriented programs may be more effective than the more common drill and practice CBI.

We found that the majority of students we interviewed believed they were more in control of their own learning when learning on computers, and that the majority of teachers we interviewed believed that they gave students more individual attention and were more aware of individual student's performance in computer classrooms than in regular classroom settings. Such findings corroborate those of our open-ended interviews and data collected in classroom observations, and suggest that the computing environments we visited were more supportive of individualized and student-centered learning than regular classrooms in those schools. We found that the majority of both the students and the teachers we interviewed believed that learning on computers was less threatening and more interesting than regular classroom learning, that discipline was less of a problem, and that students better understood the material presented during computer based learning. These results again corroborate those of our open-ended interviews, and suggest that the educationally disadvantaged students involved in the Computer Pilot Program were less threatened and more motivated by computer based learning. We found that the majority of students we interviewed believed that they had greater control over their own learning when learning on computers. These results corroborate the results of open-ended interviews and are themselves corroborated by the findings of the classroom observations. Finally, we observed nearly twice as much cooperative learning taking place among students involved with CBI as taking place among the same students involved in regular classroom instruction. Such finding suggests that the environments of the computer rooms we visited were more cooperative than those found in regular classrooms in the same schools.

Our results thus indicate that within New York City's Computer Pilot Program, the use of comprehensive CBI is altering classroom cultures to create learning environments which are more student-centered and cooperative, where teachers are more the facilitators of learning and learning is more individualized, and in which students are less threatened, more motivated, and have greater perceived control over their own learning. That classroom environments are indeed being altered in these ways is particularly demonstrated by the fact that in most cases classroom observations involved the same students and teachers. Moreover, our findings linking increased academic performance to students' participation in the Computer Pilot Program, suggest that such changes were in some sense supportive of student learning, at least among the educationally disadvantaged student population so involved. Indeed, it is our belief that the success of the CBI programs we evaluated derives at least in part from the effects of such environments on the students involved in them. Future research will investigate such effects more fully.

References

Bangert-Drowns, R. Kulik, J. & Kulik, C-L. (1985) Effectiveness of computer based education in secondary schools. *Journal of Computer-Based Instruction, 12* (3), 59-68.

Bork, A. (1985) Children and interactive learning environments. In M. Chen & W. Paisley (Eds.) *Children and Microcomputers*. New York: Sage Publications, 267-275.

Chambers, J. A. & Sprecher, J. W. (1980) Computer-assisted instruction: current trends and critical issues. *Communications of the ACM, 23*, 332-342.

Electronic Learning Laboratory. (1982) *On Task Behavior of Students During Computer Instruction VS. Classroom Instruction*. New York: Teachers College, Columbia University.

Jamison, D., Suppes, P. & Wells, S. (1974) Effectiveness of alternative instructional media. *Review of Educational Research, 44*, 1-67.

Kulik, J. (1981) Integrating findings from different levels of instruction. Paper presented at the annual meeting of the American Educational Research Association, Los Angeles.

Kulik, J., Kulik, C-L. & Bangert-Drowns, R. (1985) Effectiveness of computer based education in elementary schools. *Computers in Human Behavior, 1*, 59-74.

Niemiec, R. & Walberg, H. J. (1987) Comparative effects of computer-assisted instruction: a synthesis of reviews. *Journal of Educational Computing Research, 3* (1), 19-37.

Papert, S. (1980) *Mindstorms.* New York: Basic Books.

Ragosta, M. (1982) *The ETS Survey of Computer Assisted Instruction.* Princeton, NJ: Educational Testing Service.

Schmidt, M., Weinstein, T., Niemiec, R., & Walberg, H. (1985) computer based instruction with exceptional children: a meta-analysis of research findings. Paper presented at the annual meeting of the American Educational Research Association, Chicago.

Swan, K., Mitrani, M., Guerrero, F. & Schoener, J. (1989) Honing in on the target: the use of CBI with educationally disadvantaged students. Paper presented at the Annual Meeting of the American Educational Research Association, San Francisco.

Taylor, R. P. (1980) *The Computer in the School: Tutor, Tool, Tutee.* New York: Teachers College Press.

Walker, D. F. (1986) Reflections on the educational potential and limitations of microcomputers. In T. R. Cannings & S. W. Brown (Eds.) *The Information Age Classroom.* Irvine, CA: Franklin, Beedle & Associates, 244-252.

White, M. A. (1986) Synthesis of research on electronic learning. In T. R. Cannings & S. W. Brown (Eds.) *The Information Age Classroom.* Irvine, CA: Franklin, Beedle & Associates, 17-20.

Multimedia, Hypermedia, and the Writing Process

Stephen Marcus
University of California

Abstract

"Knowledge is 'know business' and in the know business there's no business like show business." (Aaron Marcus, Interface Design Consultant)

This presentation deals both with know business and show business, presenting and discussing some of the current and evolving technologies that integrate graphics, sound, animation, text, and video. These tools are challenging our notions of what it means to consolidate, represent, and present what we know. In particular, they are affecting how we use writing and what we have come to expect form it.

The session will suggest research questions, curriculum design criteria, and instructional strategies related to the teaching of writing in these high tech environments.

Included will be attention to hypermedia environments for the Apple II family of computers and a status report of a *HyperCard* project for English/ language arts supported by Apple Computer's Office of Educator Training.

Exploring Art and Technology Using LEGO TC Logo

Cathy Helgoe
LEGO Dacta

Eadie Adamson
The Allen-Stevenson School, New York, New York

Abstract

In this presentation we will describe a curriculum development project for fourth- through eighth-grade students. The project focused upon the connections between art and technology. Using videotaped segments of kinetic (moving) art works as inspirational examples for individual projects, students built kinetic, light, and computer controlled sculptures.

Our educational objectives for the course were as follows:

1) To appeal to students turned off by typical science and technology activities. An art approach to computer rather than a technology, science, or math approach can attract students who otherwise might not think of exploring science, math, and computer environments.

2) To expand the view of those using computers and in a strictly utilitarian way. A hands-on experience grappling with "artistic" design issues can deepen students' and teachers' understanding and appreciation of applied science and math tasks.

Students used LEGO TC logo materials as well as other materials to build and program their sculptures.

Integrating the Computer, Writing Development, and Self-esteem in the Middle School Child

Mary S. Haverfield
Gwinnett County Public Schools
Snellville, Georgia

Abstract

This session outlines the use of a computer in the development, refinement, and final production of a booklet authored by seventh-grade language arts students. Word processing is used to draft stories, poems, and factual writings. Desktop publishing integrates and formats the final drafts into an attractive, illustrated booklet. The computer is a central element used for motivation and interaction for building ideas and concepts related to the production of the booklet. Group interaction is developed through brainstorming, peer teaching and evaluation, and student/teacher writing conferences. Participants can gather information necessary to start a similar project in their own classrooms.

New Initiatives in Technology Education

Martha Hancock
Jefferson County Public Schools
Louisville, Kentucky

Abstract

Jefferson County Public Schools, in Louisville, KY is implementing a 3-year family-focused project entitled "N.I.T.E. (New Initiatives in Technology Education) School." In this program, 12 of the county's middle schools (grades six through eight) are utilized after traditional school hours as centers for 10 to 15-year-old students who will work with their parents or adopted "guardians" from the business community to design technology-based instructional materials. The proposal was developed in response to two demonstrated needs in the Jefferson County community: a need for increased parental involvement in the education of students in the middle years, and a need for higher student achievement in reading, mathematics, and critical thinking skills at the middle school level. It also recognizes an emerging societal concern—the need to retool members of the labor force whose skills and capabilities must be regularly upgraded to meet the growing demands of our technically-oriented economy. This can be accomplished by expanding the services of public education and the use of educational facilities.

Families, including those of disadvantaged backgrounds, are participating in the 8-week evening training courses that will be offered at these centers. Local businesses were solicited to involve their employees as guardians in the project by adopting students in a mentoring relationship and accompanying them to the training sessions. During these sessions, students and their parents or guardians develop computer based instructional materials using interactive multimedia technology. Students utilize materials in their classes and teachers receive training to encourage the integration of these products into their instructional planning. Take-home computers are available to the participants during their involvement in the project to provide more opportunities for students and their families to work together. The products developed by the participants are disseminated throughout the county, the state, and the nation via the Clark Foundation Project, a network of urban school districts who serve the disadvantaged, locally sponsored institutes, conference presentations, and the district's electronic participation in the Apple Global Education Network and Worldlink.

This project has yielded a variety of positive outcomes, including the development of a replicable program in which students and community members cooperate in the use of new learning media. Significant gains in the technical and academic skills of both students and members of the local work force should result, as well as increased parental and community involvement in the education of middle level students.

A Hard Drive at Risk: An Apple "Equal Time Grant" Winner

Carol Goltz
West Anchorage High School
Anchorage, Alaska

Abstract

Learn about a high-tech, high-touch project. A cross-curriculum, team approach to grant writing resulted in a Macintosh Lab at West Anchorage High School. Discover how students use Macintosh computers, printers, CD-ROM players, LaserDisc players, and software to improve their skills in language arts and science. See how cooperative learning and cross-age tutoring have resulted in improved attendance and behavior patterns, better grades, higher self-esteem, more positive attitudes toward teachers and schools, and increased collaboration with peers. Hear what students are doing with *HyperCard* and interactive media. Learn about a project involving a high school, an elementary school, and a university—a project that is making a real difference for at-risk kids!

What Exactly Do We Mean By "Logo-Like"?

Gary S. Stager, Moderator
Director of Training: Network for Action in Microcomputer Education
President of ISTE's SIGLogo

Abstract

Much attention has been recently paid to the discussion of whether or not a specific type of tool, application, or instructional piece of software is "Logo-like." This panel hopes to clarify what people who use the term *Logo-like* mean when they say it. Are other software environments *Logo-like*? Is there a list of objective criteria which may be applied to determine *Logo-likeness* or are there more affective influences (pedagogy, philosophy, social interaction, teacher intervention, culture formation) that make a piece of software or educational environment *Logo-like*?

This discussion has serious implications for the way we use Logo and other educational software, articulate our educational mission(s), how Logo-users percieve the rest of the educational community, and how the educational community regard proponents of Logo use and *Logo-like* thinking, teaching, and learning.

Panelists

Glen Bull
University of Virginia

Brian Silverman
Director of Research and Development
Logo Computer Systems, Inc.

Sharon Yoder
ISTE—University of Oregon

Marian B. Rosen
Ladue, MO Public Schools

Eadie Adamson
Allen-Stevenson School, NYC

Sponsor: ISTE

T1-9 INQUIRY, COMPUTER, AND THE MATHEMATICS CLASSROOM (SCS)

Inquiry, Computer, and the Mathematics Classroom

Nancy Roberts, Chair
Lesley College
Cambridge, MA

Abstract

This session will explore the issues surrounding classroom use of a computer based environment for inquiry learning. We will demonstrate a visual programming language, Function Machines, that has been used to introduce new approaches and foster inquiry for a variety of mathematical topics.

A review of six months of classroom experiences, from grade four through high school, will be presented. Issues that arose in our attempt to use technology to support inquiry learning in mathematics will be discussed.

Panelists

Richard Carter
BBN Laboratories
Cambridge, MA

John Richards
BBN Laboratories
Cambridge, MA

Cornelia Tierney
Lesley College
Cambridge, MA

Sponsor: SCS

Accreditation Issues Dealing with Computer Ethics

Terrell Ward Bynum
Southern Connecticut State University

Joseph Turner
Clemson University

Walter Maner
Bowling Green State University

Abstract

Accrediting bodies for computer engineering, computer science, and information science have recently considered anew the issue of requiring a "social and ethical" component in college degree programs. What are the most recent recommendations and requirements, and what rationales are offered to support them? These are the questions to be addressed by the main presenter, Dr. Joseph Turner, who is Chair of the Computer Science Department at Clemson University, as well as Chairman of the Computer Science Accreditation Commission. Responding will be Dr. Walter Maner, Director of the Artificial Intelligence Laboratory at Bowling Green State University. Time will be reserved for questions from attendees.

Back to the Future: Telecommunications Beyond Motivation

Robert Spielvogel, Chair
LEARNING LINK National Consortium
New York, NY

Abstract

Through a panel discussion commercial and educational telecommunication experts will address the questions: Where is the educational value in telecommunications? Is there something beyond motivation?

Discussion will focus on issues of training, access, equity, proper use, service, management, and standards.

The panel through its breadth of representatives—from Prodigy to FrEdMail—will debate all sides of the issues. The hope is to move beyond the fun of telecommunications and explore why it is necessary to embrace this application.

The explicit purpose is to focus on those uses of telecommunications that have valid and unique applications to education.

Panelists

Al Rogers
FrEdMail
Bonita, CA

Robert Gehorsam
Prodigy
White Plains, NY

Martin Harris
PrevNet
Sacramento, CA

Ellen Chaffin
LEARNING LINK/WNET
New York, NY

Bobby Kurshan
Roanoke, VA

Martin Engel
Apple
Cupertino, CA

Linda Roberts
Office of Technology Assessment
Washington, DC

Sponsor: SCS

150

National Geographic's Kids Network

Marti Cantrell
Pi Beta Phi Elementary School
Gatlinburg, Tennessee

Abstract

National Geographic's Kids Network will be discussed and demonstrated by Marti Cantrell, an elementary computer teacher from Gatlinburg, Tennessee. For the past three years, Mrs. Cantrell has worked with a fourth-grade classroom teacher in implementing the science/telecommunications curricula from the Network, and she has ideas, suggestions, and comments to help participants understand what Kids Network is all about. Practical topics will be discussed to assist those who want to utilize this innovative teaching technique in their own classrooms. Besides demonstration, discussion, and questioning, a descriptive video showing students working with the Network will be viewed and handouts will be provided.

Hello!, a unit to help children learn about telecommunications, provides opportunities to use data gathering techniques as children from all parts of the world compile pet data from their classroom and report via modem to their teammates. Acid Rain engages students in the vital issue of pollution as they study stream and rain samples from their area and report their data to teammates in other parts of the world. For both units, a well-known unit scientist helps the students, by way of letters written and mailed electronically through computers, to compile and study the data, suggesting pertinent topics for discussion and pointing out trends in the total network data. Both units are very well suited to be integrated into a science curriculum, and Mrs. Cantrell will have suggestions for classroom teachers who want to see their students participate in this valuable science experience, using a computer and modem to team up with other investigators around the world. The software for both of these units will be demonstrated, as well as other materials needed to participated in the network.

Values in the Computer Science Curriculum

Terrell Ward Bynum
Southern Connecticut State University

Keith Miller
The College of William and Mary

Abstract

What is the best way to integrate the study of ethics and values into the undergraduate computer science curriculum? Several methods have been tried or suggested at various universities around the country. These include: (1) offering a whole course, entitled for example "Computer Ethics" or "Computing and Society"; (2) integrating a module of computer ethics into almost every undergraduate course in the computer science curriculum; and (3) including significant ethical and social components in a senior "cap-stone" software engineering project. This session will consider each of these three approaches, with major emphasis upon the first two. Textbooks will be available for attendees to examine and a bibliography and sample syllabi will be handed out. Time will be reserved for questions from attendees.

The Role of Technology in the Restructuring of Schools

Tony Jongejan, Chair
Western Washington University
Bellingham, WA

Abstract

During the 1980s, the rate change from an Industrial Age to an Information Age and the accompanying change from blue collar jobs to white collar jobs increased dramatically. Yet our school systems have not adapted in any significant way to these changes and are ill-prepared to prepare student for the 1990s. If today's students are to function effectively in tomorrow's world, they need a different type of schooling from the one we went through.

We have little idea what the tools of the Information Age will be in the future. Yet our students must be able to adapt to the changing nature of the tools available to them. And, although we are aware that the amount of information available to students is increasing rapidly, we have made little headway in implementing the needed changes in the traditional methods of accessing it. At the same time, significant subgroups of our population are not being served by our schools.

So, once again the call for reform in education is before us, but this time reform may not be enough. Thus, a major component of the reform efforts of some states such as Massachusetts, Arkansas, and Washington and school districts such as Dade County, Florida; Rochester, New York; and Santa Fe, New Mexico includes some form of school restructuring. The literature characterizes restructured schools by reforms in five general areas: learning experiences, teachers, curriculum, organizational leadership and structure, and governance and funding.

The role to technology in all of this is intriguing. From one point of view, it is partly because of technology that these changes are happening. From another point of view, it is because of these changes that we need technology. Yet, it may very well be that taking advantage of available technologies will be the corner stone in the restructuring process.

Thus, as technology using educators, we need to be aware of our role and the role of technology in relation to the call for restructuring our nation's schools. This panel will focus on the importance of teachers and the role of technology in the restructuring effort.

Panelists

Alan November
Wellesley Middle School
Wellesley, MA

Bob Pearlman
American Federation of Teachers
Cambridge, MA

Doris Ray
Maine Computer Consortium
Auburn, MA

Sponsor: ISTE

Macintosh Lab for the Visually and Hearing Impaired

Janet Fleharty
Colorado School for the Deaf and the Blind

Abstract

Visually impaired and hearing impaired students at the Colorado School for the Deaf and the Blind are using a Macintosh Lab to build communication skills, to become familiar with mouse and menu techniques, and to learn a variety of software programs. Fourth through sixth graders practice their reading and writing skills by producing stories and books on the computer and communicating with their sighted and hearing peers via telecommunications software and a modem. The blind students use word processing programs that are either voiced or in large type. Multihandicapped students fill out job applications, write resumes and simple memos, and design personal letterhead stationary. An art class explores new modes of creativity. An 11th- and 12th-grade hearing impaired class produces a school newsletter using a desktop publishing program.

This session will deal with practical suggestions on managing a computer lab including setting it up, scheduling, training staff, and coordinating activities with existing curriculum. Specific useful software will be demonstrated as well instructional ideas that were successful. The Macintosh computers are part of an educational grant from Apple, Inc.

High School by Satellite in Kentucky: Live and Interactive!

Leslie Flanders
KET, The Kentucky Network

Abstract

KET, The Kentucky Network, has found distance learning's perennial problem of keeping teachers in touch with students at a distance to be an opportunity for invention. In the physics and math (Probability/Statistics and Discrete Math) courses taught to high school students via satellite each student has a wireless keypad that he uses to respond to the teacher's questions. In less than four seconds a dynamic graph becomes apparent to the teacher in the television studio representing the total responses. The teacher in turn can show the graph to the students over the television.

This presentation will focus on the courses KET is delivering and the many types of student/teacher interaction involved, particularly the interactive keypad system.

Learning "The Write Stuff" Through Computer Technology

Debra Freedman
New York City Board of Education
Polytechnic University
Kingsborough Community College

Jerry Stimmel
New York City Board of Education

Abstract

"Tell me, I forget—Show me, I remember—Involve me, I understand." Handicapped and speech impaired students have a unique problem. They grow up in a world where they are sometimes dependent upon others. Psychologically, this is often a greater handicap than the handicap itself. By using the computer, the students are able to experience a feeling of active participation. Listening, speaking, writing, and reading are part of the communication skills that we want to teach. This is done through the computer.

The object is to combine the *Total Physical Response* (auditory, visual, kinesthetic) with the use of the computer. The project is being implemented at an intermediate school in Brooklyn, New York. There are approximately 60 to 65 students in the speech program. Under this category are children with retarded mental development, neurologically impaired, physically handicapped, and learning disabled.

Logo was first used to introduce the students to the computer. Since the language is user-friendly it gave the students a feeling of confidence, self-esteem, and accomplishment. The students called themselves "The Speech Wizards." Through the use of word processing, The Speech Wizards were able to create a publication called *BITS, BYTES AND PIECES*. The computer encouraged The Speech Wizards to write, and we acquired pen pals in New York State, California, Japan, Holland, and Israel.

Computer technology is useful in the design and implementation of a language curriculum. It allows for the use of software to teach verb tenses, using a technique students can learn while thinking it is a game. Teaching is more diversified and individualized. Discipline problems are reduced to a minimum due to the students' involvement with the computer.

The students progress is demonstrated by informal and formal tests that indicate an increase in their communicative language skills. Using computer technology is a challenge, not only to the students, but to the professionals involved in the educational field.

Using LogoWriter to Enhance Learning of Elementary French

Eadie Adamson
The Allen-Stevenson School
New York, New York

Abstract

How can you use computers for beginning French students with a very limited vocabulary in ways that will be meaningful and will augment their learning? What can you do if you do not want to use computer games or drill and practice programs? For the past two years we have been using French *LogoWriter* with our sixth-grade French students. Taking advantage of the fact that these students are already familiar with *LogoWriter*, we have introduced the foreign language version in the context of the beginning French classes which are co-taught by the French teacher and the computer specialist. During the course of this project we have made some interesting observations about the effects upon learning and motivation and have discovered some important advantages for the teacher. This presentation will discuss the manner in which the program is used, demonstrate some of the activities that have been introduced, and comment on the impact this of this approach, based on both teachers' observations. A handout will include procedures for a French *LogoWriter* hypertext tool to turn a "dictionary" page into an interactive dictionary, providing the student with the English translation of a selected French word.

If This is Tuesday, This must be Belgium...Enhancing Foreign Language/Computer Skills through the Use of Spanish and French LogoWriter/Telecommunications

Leon Foster
Penfield Central Schools
Penfield, New York

Abstract

In a cooperative project between language and computer education departments, students at Bay Trail Middle School in Penfield, New York conduct research via an online information service, create text and graphics projects using *LogoWriter*, and telecommunicate those projects with students in other schools.

Students write in English, Spanish, French, or German, then illustrate their projects through graphics programming in Spanish and French using the Spanish and French versions of *LogoWriter*.

Language skills are enhanced through consistent use of writing and programming in the language outside of the language classtime. Simulation programs and learning games in French and Spanish round out computer use to build written and oral language skills.

This presentation will include examples of student work and curriculum materials.

Using Technology to Enrich the Whole Language Classroom

Peggy Healy Stearns
State University of New York at Buffalo

Abstract

Interest in whole language is growing rapidly. This presentation will explore ways in which technology can support and enhance the activities of the whole language classroom.

As we review the goals associated with the whole language approach, the role of technology becomes clear. The whole language teacher attempts to create authentic situations in which reading, writing, listening, and speaking have purpose--in other words, situations in which there is meaningful content and a real audience. The whole language classroom is rich in opportunities to read, write, listen, and speak. It is a classroom in which the learner is empowered.

These are goals that can be supported by the increasingly powerful communications tools now available to students at even the elementary level, tools that provide opportunities for creative expression and meaningful presentation and publication for real audiences.

The presenter will suggest a wide range of appropriate software and describe specific activities that support the whole language approach as well as promote the development of cooperative learning and critical thinking skills.

Establishing Accreditation Guidelines in Technology for Teacher Education Programs

Lajeane G. Thomas, Chair
Louisiana Tech University
Ruston, LA

Abstract

The International Society for Technology in Education (ISTE) has recently become a constituent member of the National Council for Accreditation of Teacher Education (NCATE), which accredits professional education units in U.S. colleges and universities that prepare professional educators to staff school programs for children and youth. As a member, ISTE is eligible to recommend guidelines or standards for Teacher Education programs in the area of computer/technology education for use in the accreditation process. ISTE is also in a position to influence standards of technology competence for other teacher education programs (i.e., science education, elementary education, business education, education administration, etc.).

The development process for accreditation guidelines requires the member organization to present evidence showing that research, practice, and expert review are used in obtaining consensus of the ISTE membership. Audience participation in this session is one of the opportunities provided for ISTE members and ave input on standards/guidelines for teacher preparation programs.

The ISTE Accreditation Committee surveyed the research on technology in restructured schools and conducted a study to identify the technology competencies needed for schools of the 90s. Research results will be presented as stimuli for audience participation that will follow. Participants select a table based on their interest or expertise in the discussion topic identified for each table. Discussion topics center on guidelines for teacher education programs in computer/technology education and recommendations for technology competencies in other teacher education programs. Ideas generated will be recorded on transparency and presented by a representative from each table.

Presenters

C. Dianne Martin
George Washington University
Washington, DC

Gary Bitter
Arizona State University
Tempe, AZ

Sally Sloan
Minneapolis Public Schools & Winona State University
North Oaks, MN

Don Knezek
Education Service Center—Region 20
San Antonio, TX

Sponsor: ISTE

Reflections on a Decade of Logo Use in Schools

Gary S. Stager, Moderator
Director of Training: Network for Action in Microcomputer Education
President of ISTE's SIGLogo

Abstract

This distinguished panel of long-term Logo users will reflect on the evolution of Logo use, integration, and implementations in the 1980s in order to share a vision of Logo's role in the future of educational computing and educational reform.

Panelists

Marian B. Rosen
Ladue, MO Public Schools

Michael I. Tempel
Director of Educational Services
Logo Computer Systems, Inc.

Linda Polin
Pepperdine University
Culver City, CA

E. Paul Goldenberg
Senior Scientist
Educational Development Center, MA

Tom Lough
Curriculum Director—LEGO dacta
Enfield, CT

Sponsor: ISTE

An Analysis of Administrative Computer Use By Secondary Principals In Kentucky

D. W. Witten
Nelson County High School, Bardstown, KY

M. D. Richardson
Assistant Professor/Coordinator of Educational Administration
R. L Prickett
Assistant Professor
Department of Educational Leadership
Western Kentucky University, Bowling Green, KY

Abstract

This study was undertaken to determine the status of computer use by rural secondary principals in the Commonwealth of Kentucky. A review of the literature indicated that very little research had been conducted nationwide regarding administrative computer use in the nation's schools. A questionnaire was developed and sent to all secondary school principals in the Commonwealth. The data were collected and analyzed in regard to computer knowledge, training, commitment, plans, and computer use by secondary principals.

The results revealed a majority of Kentucky's secondary school principals are woefully uninformed and poorly trained to use computers in the management of their schools. As a result, computers were, at best, being used haphazardly and inefficiently. A correction procedure recommended was mandatory training and inservice for both existing and future secondary principals.

Introduction

In recent years, computers have been used successfully in the effective management of most businesses. Since secondary schools are some of the largest businesses, the logical assumption is that principals would be effectively and efficiently using the computer as a management tool. This study focused on the knowledge of rural Kentucky secondary school principals regarding computers as well as the extent to which the principals use computers for administrative purposes in their schools. In a period of time when principals are facing severe budget limitations due to the lack of funding from the state, they are called upon to manage the resources received in the most efficient manner to provide the best education possible for the students.

Background

Investigation revealed very little research available on a national basis to measure the use of computers for administrative purposes in rural secondary schools. Research has been and was being conducted regarding computer use in the classroom, or at the district-office level but not in the day-to-day management of schools (Spuck & Anderson, 1983). In the absence of valid data, there is a need to determine exactly what building principals in rural areas did or did not know about computers and how they were or were not using computers to assist in the efficient management of their school. Therefore, a survey questionnaire was designed specifically for this study and was mailed to all secondary principals in Kentucky (N=297). Since Kentucky schools are predominantly rural, the results should be a reflection of non-urban responses. However, for validation, responses received from the two urban areas of the state were removed from the final results of this study. Consequently, responses were received from 154 rural secondary principals, for a 51% response rate.

Computer Use

Computers have been widely used in the classroom for educational purposes, but their use for administrative functions in most schools has received limited attention. The extent to which schools use computers for administrative purposes depends on the principal's level of computer literacy (Kearsley, 1988). Computer literacy or microliteracy is defined as the awareness of, knowledge about, and programming experience with microcomputers. Microliteracy must precede the effective and creative use of these machines as an administrator tool, a teacher tool, and a student learning tool. The degree of microliteracy an administrator possesses is largely due to the perceptiveness of the individual as well as the ability to effectively change methods of work as new technology is developed.

While many rural school districts have jumped on the computer bandwagon, most of the purchases have been for instructional purposes, not administrative uses. Administrative purchases of the latest technology have lagged behind the instructional uses for a variety of reasons. Typically, rural school principals have adopted the posture that if it works, don't fix it (Anderson, 1987).

According to Johnson (1985), one of the primary factors in the administrative use of computers is the principal's ability to accept change. Acceptance of computer technology cannot be expected unless consideration is given to each individual administrator's ability to handle change. Although most principals would admit work in the school office is usually backlogged, they are still hesitant to propose radical changes in the day-to-day operation of the school.

According to Pogrow (1985), paperwork is the most mismanaged opportunity in education. Computers have the potential to reduce much of this paperwork, by 50 to 90 percent in many situations. These improvements would allow for a complete return on the investment of computer hardware and software within months.

Using the old start-sharing computer systems, many principals experienced prolonged delays in receiving basic printouts which were needed to manage day-to-day activities. Printouts arrived either too infrequently to be of any use, or too late for administrators to take advantage of the information provided. Therefore, principals were forced to keep their own set of inefficient manual records in order to manage these daily activities which ultimately led to both fear and suspicion of the computer.

According to Pogrow (1985), The ideal school administrative computer system contains the following elements:

1. Data entry is easy, quick and is done as the transaction are occurring.
2. The computer checks for errors at the time of data entry and makes it easy to immediately change the information.
3. Changes can be entered into the computer easily and the current status of student enrollments are updated instantly with the results easily accessible.
4. Common information is automatically transferred across applications.
5. In schools with more than 500 students it is possible to have several linked computers or terminals so that people in different parts of the building can be working on the same system at the same time.
6. Principals and staff without technical backgrounds should be able to easily ask basic questions of the stored information and obtain immediate results.
7. Principals and staff without technical backgrounds should be able to easily design report formats in minutes.

Hardware and Software Selection

While it is a huge task to evaluate a student information system adequately, according to Anderson (1987), an administrator cannot afford to choose one without evaluation. The excessive costs of an inadequate student information system could be very damaging to the school district and the careers of administrators and/or board members who might have championed such a system. It is possible to muddle through with an outdated and inefficient information management system but why take the chance? The principal must be aware of the obsolescence factor. He/she should not purchase computers and software without a great deal of forethought and consideration. Crucial to the success of any purchase was the evaluation of the questions "Where are we?" and "Where do we want to go?" The principals who can answer those two questions will be able to decide how they are going to get there and how they will know when they have arrived (Johnson, 1985).

The first thing that a principal should consider is that all school offices are different. Each office has its own routine and certain strengths and weaknesses in its personnel. The most important question a principal could ask was: "Who requires what information, when, and where?" According to Anderson (1987), the principal must consider the following:

1. How much time does it take to search for information?
2. How much time would it take to input the information needed to provide those reports touted as useful by the computer salesperson?
3. Will verifying and validating the information be a time consuming process?
4. How much time does my staff spent converting data into information I can use as a principal?
5. Can the distribution of the information be supported by computer technology?

There are two components to every computer system—the hardware and the software. The administrator should find out what vendors could assist in the software selection or design and the resulting need for computer hardware. Administrators should always consider the prospective vendor's reputation (Cantrell, 1989).

Anderson (1987) provides these software selection guidelines:

1. The software that pleased principals the most usually came from designers who were school administrators first and microcomputer designers second.
2. Companies that had been producing administrative software for longer periods of time usually had programs with fewer errors.
3. A company that was producing well-designed and well-written software should willingly provide a number of customer references.

Good student information systems consist of several integrated modules. Common modules are student data base, attendance, accounting, grade reporting, and class scheduling for secondary schools. Attendance modules bring to attendance accounting an excellent audit trail and a sense of data integrity that has previously been non-existent. Attendance may be checked by period using an optical scanner and cards for each student.

161

Repondents Who Use a Computer at Home

Computer User	N	Percentage
Yes	21	13.64%
No	133	86.36%
Totals	154	100.00%

Table I.

Many systems had additional modules that may have included test scoring and analysis, test history, and instructional materials. Large, rural secondary schools usually require a very efficient class scheduler. As such systems may include the ability to perform different functions, it is important to determine the value of these additional functions to the district when selecting a student information system (Anderson, 1987).

Data entry is a time consuming and relatively expensive task which could be expedited by the integration of modules so the basic student data must only be entered once. Of course, school district requirements could vary substantially by state, county, student population, grade level, special schools, and unique district reporting requirements. Other factors to explore include user inquiry capabilities, software flexibilities, maximum school size supported, quality of program code, programming documentation, hardware reliability, video screen response time, quality of vendor support, and the availability of countless other important features. Purchasers must know what to look for before they go shopping (Anderson, 1987).

The Vision of the Future

Rural principals are trying to thrive and survive with the latest rage of the technological age, the computer (Crawford, 1982). Rural school principals can make the difference between whether the school system will have a well-oiled machine or a bucket of bolts.

Educational administrators have to look forward to the school office of the future. Office automation is a tool rural administrators can use to better manage their institutions. In order to use office automation successfully, administrators should educate themselves about these differences in the offices now as compared to what their office of the future can become (Cantrell, 1984).

The business of education is becoming more complicated and competitive each day (Johnson, 1985). In order to be a successful school administrator, there are problems which have to be overcome. Some of those problems are (Crawford, 1985a):

1. Fear of computers or the problems a computer might cause.
2. Initial cost of hardware and software.
3. Lack of knowledge about what tasks can or cannot be performed by a computer.
4. Security of the methods needed to ensure limited access to data.

Rural principals can use computers in very creative ways. A small laptop computer can help structure the elements of clinical supervision, record data items, and most importantly, quickly produce a complete transcript of classroom interaction for the teacher prior to the post-observation conference (Kuralt, 1987).

The availability of more effective computer systems means that office automation is reaching a point where paperwork could be vastly simplified. No longer should the rural school administrator be forced to work till midnight to get reports completed. More time can be made available to deal with student needs (Pogrow, 1985).

Some states have adopted a very progressive effort to introduce computers into the management of the schools. Georgia is attempting to automate the schools of the entire state by installing the Statewide Information System (SIS). This project is slated to be completed by 1994 and will radically reform the day-to-day operations of the state's 1800 elementary and secondary schools as well as 30 other technical and vocational institutions. By 1994, the schools will be equipped with specialized applications that run on minicomputers, local-area networks and stand-alone microcomputers which will give each school access to the grades and attendance reports of more than 1.3 million students (Pang, 1989).

Experience with using a computerized management system should be an express requirement of all new administrators and counselors, whether in rural school districts or urban districts. Once a rural secondary school has good administrators who are take charge persons, properly selected computer systems could substantially improve the quality of administrative practice (Pogrow, 1985).

For a significant change like the introduction of computers into the schools to be successful, educational administrators must lead the way. They will be either the major leaders or barriers to successful computer use. With the new breed of school administrator currently being trained, the computer will become a tool to revolutionize student records and information processing.

Educational administrators are subjected to immense social pressures for the improvement of the education of

Repondents Having Any Special Computer Training

Training	N	Percentage
Yes	32	20.78%
No	122	79.22%
Totals	154	100.00%

Table II.

Administrative Tasks Related to Students

	Yes		No	
Task	**N**	**Percentage**	**N**	**Percentage**
Records	91	59.10%	63	40.90%
Scheduling	97	62.99%	57	37.01%
Attendance	79	51.30%	75	48.70%
Grade Reporting	83	53.90%	71	46.10%
Discipline	62	40.26%	92	59.74%
Career Planning & Placement	41	26.62%	113	73.38%
National Standardized Testing	58	37.66%	96	62.34%
Kentucky Essential Skills	111	72.08%	43	27.92%
Other Student Testing	13	8.44%	141	91.56%

Total N=154

Table III.

students in their institutions, particularly in rural areas. Reform of the curriculum and strengthening of teacher qualifications are currently receiving high visibility, but one of the most precious resources—time—must be conserved and managed to permit other factors to operate in improving educational results. Microcomputers, when properly used, assist administrators in saving time ordinarily consumed in routine tasks, and thus provide additional time for working directly on vital leadership functions. Improvements in leadership efficiency are not likely to be significant, however, unless the computerized administrative system has been carefully built around a comprehensive and systematic plan clearly establishing goals, alternative methods, costs, benefits, responsibilities, and schedules for the implementation of the computerized system. As technical capabilities continue to increase, cost continues to decline, and as humans improve their abilities to utilize the new technological tools, a new era in administrative computer applications appears imminent (Spuck, 1983).

Findings

The respondents in this study reported only 13.64% who owned a computer at home and only 20.78% who had any special computer training. (See Tables I and II). Responding rural principals did not use computers extensively in school management. For example, only 62.99% used computer scheduling, 53.90% used computer reported grading and 51.30% used the computer in maintaining attendance data. Only 26.62% of the responding rural principals used the computer for student career planning and placement (See Table III).

Computer use was almost non-existent in the administrative tasks related to school personnel. Only 27.27% of the responding rural principals used the computer to assist in the placement or assignment of faculty and 3.24% used the computer in recruiting faculty. Less than 2% (1.94) used the computer in teacher evaluation (Kuralt, 1987) although 17.53% used the computer to generate evaluation reports (See Table IV).

Surprisingly, only 15.58% of the responding principals were "very committed" to the administrative use of computers while 56.49% were "somewhat committed" and 27.92% were "not committed" (See Table V).

Administrative Tasks Related to Personnel

	Yes		No	
Task	**N**	**Percentage**	**N**	**Percentage**
Staff Records	31	20.12%	123	79.88%
Evaluation Reports	27	17.53%	127	82.47%
Recruitment	5	3.24%	149	96.76%
Selection	16	10.38%	138	89.62%
Placement or Assignment	42	27.27%	112	72.73%
Leave (Sick or Other)	39	25.32%	115	74.68%
Substitute and Part-time Employees	12	7.79%	142	92.21%
Observation Tool	3	1.94%	151	98.06%

Total N=154

Table IV.

Repondent's Commitment to Administrative Computer Use

Type	N	Percentage
Very Committed	24	15.58%
Somewhat Committed	87	56.49%
Not Committed	43	27.92%
Totals	**154**	**100.00%**

Table V.

Implications

In a time in which limited resources are being devoted to Kentucky's schools, it is certain that we must manage the existing resources as efficiently as possible. The rural secondary principal needs to have a firm command on the available information need to make the necessary decisions to effectively manage the school in which he/she works. Computers are the answer to many of the information management needs of the rural principal. Rural principals must actively seek the information needed to use the computer as a productivity tool.

Computer use is being widely accepted in a limited number of Kentucky's schools. This trend is one which must continue into the 1990's so that the rural secondary principals can manage the information and limited resources available.

Recommendations

The recommendations are based upon the review of literature and survey findings. First, it is recommended that the Commonwealth of Kentucky Department of Education mandate computer training for all current and future rural secondary school principals. This mandatory education or training could be provided by the State Department of Education, the universities which prepare school administrators, or the professional organizations to which administrators belong. After completing the mandatory training the rural school administrator should be able to:

1. Describe the possible administrative uses of computers.
2. Determine what applications the school would have which could benefit from computer use.
3. Select the best software and hardware for a given need.

Brand of Administrative Software Used

Brand	N	Percentage
School Master	51	33.12%
Osiris	30	19.48%
None	63	40.91%
Other	10	6.49%
Totals	**154**	**100.00%**

Table VI.

4. Develop a plan for the successful implementation and use of computer applications.
5. Use the computer as a personal productivity tool.

Second, it is recommended that the Department of Education provide direction and even standardization on the computer hardware and software to be used in the administrative areas. Until some standards are developed by the state, Kentucky's rural school administrators are likely to purchase computer software and hardware which may not meet their needs as well as the needs of the State Department of Education.

Third, it is recommended that major revenue allocations from the state be made to fund the purchase of computer software and hardware. The major obstacle to purchasing computers is the lack of money. The Commonwealth must recognize and address this need with adequate funds before there will be any major implementation of computers into a majority of Kentucky's rural schools. Funding must also be allocated for a state wide computer information system to be developed.

Fourth, it is recommended that a source of information, which rural school administrators could use to share information and to gain the necessary knowledge they need to be successful in their use of computers, must be developed. This should be accomplished by establishing a users group of rural school administrators who could meet and share ideas with each other as well as those who desire information about the proper ways to begin the implementation of an effective computer system.

Repondents Having a Plan for Administrative Computer Training

Plan	N	Percentage
Yes	56	36.36%
No	98	63.64%
Totals	**154**	**100.00%**

Table VII.

Fifth, it is recommended that an electronic bulletin board for Kentucky's rural school administrators be established to allow principals to share ideas about their successful uses on computers. This service could also be a source of information that would help rural administrators solve various problems they were having with their computer software or hardware. The service must be easy to use and one which the State Department of Education would provide as a resource tool to the school administrator.

Sixth, there is a need for continued research into the developments that should be taking place in the future of computer use in the administrative areas of

Kentucky's rural schools. This study should be repeated as well as expanded upon. Research should be conducted in the area of elementary school administrative use. Further, research should be conducted which would compare schools of different sizes and types as related to computer use. Additional research would also be beneficial in determining how successful rural school administrators have used computers in the administrative areas of their schools.

When comparing the newness of computer technology to the age of most principals, it was obvious that most principals have received no formal training in the use of computers. Typically, principals who did not understand the value of technology are somewhat hesitant to actively seek continuing education in the area of computers. By human nature people are resistant to change and scared of exploring the unknown.

Conclusions

Based upon the research, a majority of rural secondary principals in the Commonwealth of Kentucky are not using computers to assist them in managing schools. These principals received little or nc formal training in the use of computers. Beyond this, the school system in which they work does not offer any type of training in the use of computers to help them manage schools. Also very evident was the lack of planning and commitment to computer use in the administrative areas of Kentucky's rural schools. Most disturbing was the lack of commitment to computer use in the area of administrative school management.

Basically, the data collected led to several conclusions. Kentucky's school administrators are in need of training, preparation, and knowledge regarding the effective use of the computer as a management tool. There is a need to assist principals in recognizing the importance of administrative computer usage as a good management and personal productivity tool. The obvious lack of knowledge and commitment to learning about computers was a startling fact.

Bibliography

Anderson, J. (1987). Sewing up the best information system. *The School Administrator*, *44*(8), 8-10.

Johnson, D.R. (1985). The office of the future. *American School and University*, *58*(1), 24-32.

Kearsley, G. (1988). What should today's school administrators know about computers? *Technological Horizons in Education Journal*, *15*(3), 65-69.

Kuralt, R.C. (1987). The computer as a supervisory tool. *Educational Leadership*, *45*, 71-72.

Pogrow, S. (1985). Administrative uses of computer: what is the ideal system? What are the trends? *NASSP Bulletin*, *69*(485), 45-53.

Spuck, D.W. & Atkinson, G. (1983). Administrative uses of the microcomputer. *AEDS Journal*, *17*(1), 83-90.

Principals' Perspectives on Computerized Student Information Management

Dr. Peter Wright
Department of Adult Career and Technology Education
Faculty of Education
University of Alberta, Canada, T6G 2G5
Telephone (403) 492-5363 Fax (403) 492-0236

Abstract

The implementation of computerized information management at the school level can be a complex and challenging task, particularly if the school is part of a large district. Success or failure depends critically on the selection of the software system. Following a lengthy process of testing and evaluation, Edmonton Public Schools recommended a single microcomputer based software system for use at both the junior and senior high school levels. Following a review of the evaluation process, this paper goes on to relate some first impressions with the use of the recommended system as seen through the eyes of the school administrators.

(*Keywords*: administrative computing, instructional support systems, implementation, evaluation, high school.)

Introduction

In the last fifteen years, computers have not only increased in power and performance, they have become more affordable and easier to use. Tasks which could once only be addressed by data processing professionals using mainframes can now be carried out by innovative 'end users' with microcomputers. Educational leaders are prominent among those who expect their enterprises to benefit from the application of computer and related technologies. In discussing curriculum trends of the future, Ornstein (1989) more directly affirms that "a school that does *not* emphasize technology will miseducate students for the future". It seems fair to state that, despite being the focus of a great deal of attention, visions for the exploitation of computers in education have been slow to materialize—profound and effective applications are far from being widespread. So how can a school go about achieving this emphasis on technology ? One strategy is for the school principal and his assistants to lead by example. Miller (1988) supports this notion in stating that "administrators' verbalizations to their staff about the virtues and advantages of computer use may well fall on deaf ears, unless the staffs can observe that administrators themselves practice what they preach". Bosch (1988) is more explicit in stating that "the ability of school administrators to raise their productivity and increase the computing capabilities of their schools is critical in the Information Age". The implementation of microcomputer based student information management provides an excellent opportunity to demonstrate such leadership and in areas which have profound impact on the teaching and learning environment.

After briefly describing Edmonton Public Schools' approach to the evaluation and selection of computerized student information management systems, this article goes on to focus on some of the outcomes of implementation in ten of its high schools as perceived by the school principal and/or his assistants.

System Testing and Evaluation

A number of factors combined to render the implementation of school based student information management systems in the high schools as a very high priority in the district's distributed systems plan. Significant among these factors was widespread dissatisfaction with services provided through the centrally located and administered mainframe computer (associated primarily with the use of outdated computer programs) fuelled by the perception on the part of a number of principals that school based microcomputers were the way to go. In short, junior and senior high school principals expected that school based computer systems would provide them with:

- Better control over information and processes in general
- Improved capability to meet student needs
- Increased administrative efficiency and effectiveness

Student information management systems are sophisticated applications which will almost certainly have profound and long term impact on the operation of a school, it is important, therefore, that their selection result from a well conceived evaluation process. The evaluation of instructional software has evolved to the point that monographs and books have been written on the topic (Owston, 1987 and Bennett, 1987 for example). The methods for evaluating instructional software, however, are largely unsuitable for the evaluation of administrative and instructional support software. As Talley (1983) has noted "because this software tends to be more complex relative to much instructional software, an administrator must either acquire more computer expertise or find someone other than a salesperson to give an accurate assessment of the worth of the package." In recognition of the foregoing, a thorough yet flexible approach to the testing, evaluation and selection of student information management systems was developed and employed by

Edmonton Public Schools. This initiative, which has been described by Wright (1989) was jointly funded by the District and Alberta Education. Evaluation and testing was led by the Distributed Systems Team (DST)—a separately managed unit of what was then the district's Information Services (data processing) Branch. The Team set out to find systems which:

- worked and would work well for most schools
- were affordable
- could be effectively supported and
- would not limit future opportunity

Systems were evaluated against progressively more detailed criteria, in this way, low potential alternatives were rapidly identified and screened out. Five systems survived preliminary screening and were subjected to detailed testing in both junior and senior high school pilot sites using full, and wherever possible, live data. Of the five systems, two were minicomputer based (one running on IBM, the other on Digital Equipment Corporation hardware). The other three systems were microcomputer based and were required to run on IBM hardware. In all cases, actual testing was conducted by a programmer or systems analyst from the DST, relying very heavily on the expertise, judgement and direction of school pilot site administrators, typically, the assistant principal. A total of 18 junior and senior high schools participated in the development of a very detailed set of weighted criteria which were grouped under the following six major headings:

1) System scope and function (what a system does and how well it does it)
2) Ease of use
3) Technical merit (system design, operation etc.)
4) Support and services (after sales support and service)
5) System qualifications (credibility and viability)
6) Vendor (who stands behind the product?)

As one might expect, system scope and function (to many, the most important area) gave rise to the largest number of criteria. Weighting factors were used to define the relative importance of individual criteria and subsets thereof. All five high potential systems were evaluated against the same criteria. The criteria, and the process through which they were developed, can be referenced in an Alberta Education report entitled Selection Criteria for Integrated Student Information Management Systems (1985).

All system capabilities and functions were tested within the context of normal school operating procedures and practices and with particular emphasis on:

- data base creation and maintenance
- pre-scheduling
- scheduling
- transition to operational status, year end and semester turnover
- attendance recording and reporting
- progress recording and reporting
- report generation
- utility functions

Using a simple zero to ten point scale, the functionality of each system was scored against the detailed criteria. A score of zero was assigned when functionality was either completely unsatisfactory or when the required capability or feature was not present. Although scoring was based significantly on the outcome of testing, all information available was considered, particularly the perceptions and interpretations of school leadership staff. Upon completion of testing, scores were multiplied by corresponding weighting factors to yield weighted scores. For each criteria, or subset thereof, weighted scores can be interpreted as measures of how well the users needs are met by a particular system.

Making the Final Selection

The output from the evaluation process was a very detailed set of weighted scores for each system under consideration. While it was entirely possible to reach a final decision based on this information, a process of normalization was employed. The normalization process not only reduced the complexity of the information but provided a quick and easy method interpreting evaluation data from a variety of user perspectives through simulation. Normalization was accomplished by first totalling the weighted scores over each of the six major areas of interest and then dividing these totals by the corresponding, maximum possible weighted scores (for those areas). The normalization process resulted in a set of six performance indicators per system. These performance indicators, which always lie in the range zero to one inclusive, are fractional measures of the extent to which the evaluator's needs in the corresponding major area are met by a given system. At this stage, all that remained to be done to determine the relative suitability of systems was to combine the performance indicators with the overall emphasis to be placed on each of the six major evaluation areas. The relative suitability of a system in each of the major areas was defined as follows:

Relative Suitability = Emphasis x Performance Indicator

where "Emphasis" is a number between zero and one hundred which can be interpreted as the percent emphasis that the decision maker wishes to place on a particular area. Since, as previously stated, performance indicators are numbers less than or equal to one,

MAJOR EVALUATION AREA (OR CRITERIA)	EMPHASIS (%)	RELATIVE SUITABILITY (=EMPHASIS X PERFORMANCE)		
		System 1	System 2	System 3
System Scope and Function	45	( )	( )	( )
Ease of Use	10	( )	( )	( )
Technical Merit	10	( )	( )	( )
Support and Services	15	( )	( )	( )
System Qualifications	10	( )	( )	( )
Vendor	10	( )	( )	( )
TOTALS	**100**			

Relative Suitability of Microcomputer Based Systems to the Senior High Schools

relative suitabilities will be numbers which vary between zero and the percent emphasis number. Decision support tables similar to the one illustrated above were constructed to facilitate final comparison.

For convenience, performance indicators can be entered in the parentheses. The percent emphasis distribution shown in the above table (which must always total 100%) represents that actually used by Edmonton Public Schools. Note that while the percent emphasis placed on system scope and function was the highest, its importance was outweighed by the collective importance of the other five areas. Entries in the relative suitability column were totalled and compared to identify the most appropriate system.

With a total relative suitability of 76% for senior high schools and 83% for junior highs, a single microcomputer based system was judged to be most appropriate to the needs and requirements of Edmonton Public Schools. This system was subsequently recommended for use at both the junior and senior high school levels. Although simulations based on varying the percent emphasis distribution were completed, the recommendation would not have changed since the system judged to be the "best" exceeded all of its competitors on each of the six performance indicators.

For detailed information on the evaluation of the minicomputer based systems, the reader is referred to an Alberta Education report entitled Minicomputer Based Student Information Management Systems in Alberta Schools (1985). For detailed information on the evaluation of the microcomputer based systems, the reader is referred to an Alberta Education report entitled Microcomputer Based Student Information Management Systems in Alberta Schools (1985).

The first ten (six senior and four junior high) schools to implement the recommended system were surveyed to determine their reasons for implementation and their initial experiences with implementation. The following reports the findings of the survey.

Anticipated Benefits

As a result of the evaluation process and detailed consultations with school administrators, a number of perceived benefits of the use of school based student information management systems were identified. Table 1 summarizes the perceived importance of these potential benefits to the junior and senior high schools.

Outcomes of Implementation (First Impressions)

The largest part the survey instrument was devoted to gathering information on the outcomes of the early months of implementation from the perspective of the school principal and/or his assistants. Generally speaking, the schools were very satisfied with their use of a local system. Nine of the ten schools surveyed indicated that they were satisfied both with their use of the system and the results they were able to achieve. Of these nine, six indicated that they were very satisfied. One school (a junior high) indicated that it was very dissatisfied with its use of the system. Nine of the ten schools indicated that they were comfortable with the

POTENTIAL BENEFIT	# SCHOOLS WHO CONSIDERED BENEFIT TO BE IMPORTANT	
	Senior Highs N=6	Junior Highs N=4
Control over student demographic information	3	4
More comprehensive student information	5	4
More up to date student information	5	4
Immediate access to student information	6	4
On site scheduling	6	3
Improved student scheduling function	6	3
School based attendance (tracking & reporting)	6	4
School based mark reporting	2	2
Improved quality of information for teachers	6	3
Improved quality of information for students	6	4
Capability to produce reports when needed	6	4
Provision of better overall service to students	6	3

Table 1. Potential Benefits of Implementing a Local System

use of the system (the exception again being a junior high school). Seven of these nine indicated that they felt very comfortable. Seven of the ten schools reported that integrating the use of the system with school operations was easy. Integration was more challenging to the junior high schools two of whom reported the task to be fairly difficult. Six of nine schools who responded indicated that more time was required to support normal school operations than in previous years. This outcome is almost certainly associated with lack of experience with a local system—exacerbated by the fact that the first task undertaken was organization for instruction (scheduling)—the most complex of the functions addressed by the system. This assertion is supported by the fact that seven of the ten schools indicated that they expected the time required to support school operations in the future would be less than in previous years. In response to the question "were you better able to prepare for school opening than in previous years ?", three of five senior highs respondents said "yes" while two of the four junior highs said "no". Despite the fact that some difficulties were encountered by the junior highs, all ten of the schools responded affirmatively to the question "In retrospect, do you feel that the use of a local system for the purpose of student information is viable ?"

Both school types had expressed the hope that access to information in general would be better both in terms of timeliness and quality. Table 2 shows the number of schools who were more satisfied with access to information as a result of using the local system than with previous methods.

Were you satisfied that	Number of schools who said YES	
	Senior Highs N=6	Junior Highs N=4
access to information was easier?	6	3
access to information was more timely?	6	3
information was more accurate and current?	6	3
more student and school information was available?	5	3

Table 2. School Satisfaction with Access to Information

Were you better able to meet	Number of schools who said YES	
	Senior Highs N=6	Junior Highs N=4
the information needs of students?	5	2
the information needs of parents?	4	3
the information needs of teachers?	6	3
the information needs of administrative staff?	6	3
the information needs of support staff?	5	2

Table 3. Ability to Meet the Information Needs of Particular Groups

Schools were asked to rate their capability to meet the information needs of specific client groups through the use of a local system. Table 3 shows the number of schools who felt better able to meet the information needs of students, parents, teachers, support staff and administrators.

Hardware and software systems were installed during spring and summer in preparation for the upcoming September school opening. Following inservice training sessions and the establishment of the student records databases, the first major task to be tackled by the schools was student scheduling. School based student scheduling has always been a priority of the senior high schools. In the past, the senior high schools had been particularly dissatisfied with both the results of scheduling and their inability to perform an appropriate number of scheduling runs (primarily associated with long turnaround times between the schools and central services data processing department). Survey results indicated that although more administrative time was spent on scheduling by the senior highs than in the past, all six indicated that they were able to complete as many scheduling runs as they required. Scheduling has also been high on the priority list of the junior highs but past experiences have led them, reluctantly, to the perception that the less flexible (e.g. non semestered) organizational patterns of such schools renders them less susceptible to computerized scheduling. Survey responses indicate that some junior highs did use the local system to organize for instruction but their approaches clearly differed from those employed by the senior highs. Table 4 reports on first impressions with school based student scheduling.

Both junior and senior high schools considered that the overall scheduling outcomes achieved from their first time use of a local system were better than they would have been from using their respective former approaches. Nine of the ten schools indicated that they expected even better results from the system in the future.

Schools were asked to indicate whether certain types of problems were encountered during their use of the local system. Table 5 shows the number of schools who encountered specified problems.

As can be seen from the table, no schools encountered loss of data problems. This, together with the low level of difficulty encountered with the use of the system, was a very important outcome of implementation and a testimonial both to the software and to staff. As might have been anticipated at the senior high school, processing and printing produced problems. This will almost certainly be due to the myriad of reports that senior highs tend to require during school opening. While problems were

Were you better able to meet	Number of schools who said YES	
	Senior Highs N=6	Junior Highs N=4
overall scheduling capability?	6	2
capability to accommodate course requests?	6	1
capability to balance semester loads?	3	2
capability to accommodate late registrants?	6	2
capability to accommadate timetable changes?	5	3

Table 4. First Experiences with Student Scheduling

Type of problem encountered	# of schools encountering problem	
	Senior Highs N=6	Junior Highs N=4
hardware failures?	3	2
application software failures?	1	2
data processing time constraints?	5	2
printing time constraints?	5	2
system availability time for staff use?	2*	3
lack of staff skill/confidence?	4	3
difficulty using the system?	0	2**
loss of data?	0	0

* Five of six schools responding
** Three of four schools responding

Table 5. Problems Encountered

encountered, the severity and frequency of occurrence were low.

Discussion

The implementation of computerized, student information management at the school level was one component of a major thrust towards a distributed approach to information management in the District. The evaluation and subsequent implementation of systems was carried out under challenging circumstances both from the political and the logistical points of view. At the outset, budgetary restraint threatened to prevent the project from being undertaken at all. The most pervasive and enduring challenge of a political nature, however, related to the evaluation of systems. Specifically, a number of senior high school principals, who were very interested in the potential of the microcomputer did not see the need to wait for the outcomes of what was to be a lengthy evaluation process. The feelings of this group of administrators are consistent with the findings of others such as Isherwood and Blackwood (1988) who observe that principals have discovered that microcomputers can do most of what they want doing when they wanted it done. Their anxiety was exacerbated by the fact that evaluation initially focussed on minicomputer based systems.

Logistically, it would seem, there is never a "good time" to conduct such an evaluation project. The full scale, real time approach to evaluation that was pursued resulted in very heavy demands and duplication of effort being placed on pilot site administrators. As the outcome of evaluation became known and the decision to implement was taken, different challenges emerged. Successful implementation had to be guaranteed. It would have been simply unacceptable to commit to a new approach to organisation for instruction (and scheduling in particular) and conclude at the beginning of a school year that it didn't work. Equally unacceptable in a large school district would be the fact that the information needs of individual schools had been met but those of the district had not. This latter consideration resulted in an immediate focus on the establishment of telecommunications links between the schools and central services through which concurrency of information could be assured.

The outcomes of the survey indicate that the implementation of microcomputerized student information management at the school level was successful. In general, the level of satisfaction of the senior high schools was greater than that of the junior highs. In large part, this is due to the fact that the basic organisational structure of the senior highs (semestering) is more susceptible to computer based scheduling than that of the junior highs. While the survey results reported in this paper derive significantly from organisation for instruction, informal indicators suggest that the use of school based microcomputers for attendance monitoring and reporting was also successful (see Classroom Connections, 1986).

As with most innovations, those who take the first steps generally display great tenacity and are strongly committed to success. While the findings presented should be interpreted within this context, it is significant that a conscience check indicated that nine of the ten schools would do it again (with senior highs being strong advocates).

References

Bosch, K. A., (1988) A Microcomputer Literacy Training Model for School Administrators, *Journal of Research on Computers in Education*, 20(4), 331-338.

Bennett, R. E., (1987) *Planning and Evaluating Computer programs*, Merrill, Columbus, Ohio.

Home Grown: Edmonton Public School District's answer to the information merry-go round (1986). *Classroom Connections, Educators and IBM: Creating new learning environments.*

Isherwood, G. B., & Blacklock, T. (1988) The Politics of Computing. *Journal of Research on Computers in Education*, 20(4), 339-346.

Miller, H., (1988) *An Administrators Manual for the Use of Microcomputers in the Schools*,Prentice Hall, Englewood Cliffs, New Jersey.

Owston, R. D., (1987) *Software Evaluation*, Prentice Hall Canada Inc., Scarborough, Ontario.

Ornstein, A. C., (1989) Curriculum Trends: An Agenda for the Future, *NASSP Bulletin*, 73 (514), 37-48.

Talley, S., (1983) Selection and Acquisition of Administrative Microcomputer Software, *AEDS Journal*, 17 (1 and 2), 69-82.

Wright, P. Choosing a Computer Based Instructional Support System: An Evaluation/Selection Model. *Journal of Research on Computers in Education* (in press).

Wright, P., & Vernik, I. *Selection Criteria for Integrated Student Information Management Systems.* Alberta Education Report (1985). Edmonton.

Wright, P., and Valbonesi, P. *Minicomputer Based Student Information Management Systems In Alberta Schools.* Alberta Education Report (1985). Edmonton.

Wright, P., and Valbonesi, P. *Microcomputer Based Student Information Management Systems In Alberta Schools.* Alberta Education Report (1985). Edmonton.

Elementary Software for the Next Generation

Barbara Kurshan, Chair
Educorp Consultants
Roanoke, VA

Abstract

The "next generation" of software is still being defined, but it certainly will be tool based, fully integrated, technologically advanced, and easy to use. Moreover, it will be powerful enough to support learners not only in acquiring basic skills, but also in solving problems, developing critical thinking skills, communicating ideas and working collaboratively on multi-disciplinary projects. There is not one product on the market today that fully contains all of these features, nor is there a workstation for children that could run this software if it existed. However, many new products and several in the idea and development stages are closing in on the next generation. This panel will explore the future of elementary software from the perspective of the software publisher, the researcher, and the developer. Topics will include:

- Future products
- Current research and development
- A look at the market for publishers
- The next generation of hardware
- Development of tools for elementary learners
- Multimedia

Panelists

Beverly Hunter
Associate Program Director
Research on Learning and Teaching
National Science Foundation
Washington, DC

Peter Kelman
Publisher
Scholastic Inc.
New York, NY

Marge Kosel Cappo
Vice President
Sunburst
Pleasantville, NY

John Richards
Software Development Manager
BBN Laboratories Incorporated
Cambridge, MA

Sponsor: ISTE

Vision Test: What Our Schools Could Be Like

Robert D'Ambrosio, Chair
New York University
New York, NY

Abstract

Last year, at NECC '89, the ISTE Board established a Committee on Future Directions of Educational Technology and asked Lud Braun to chair it. During the past few months, IBM has decided to support a study by that Committee to identify ways in which technology can improve the learning experiences of precollege students; to identify its cost effectiveness; and to highlight the consequences of failing to exploit technology during the current window of opportunity (there is a currently high level of interest at all levels of our society in doing something dramatic to improve the education of our children).

The principal outcomes of the study will be a set of reports (paper and video cassette) aimed at educational decision makers at every level from the parent to district superintendents, chief state school officers, and state and Federal legislators, and intended to convince them to support increased *intelligent* applications of technology in our classrooms.

The Committee is carrying out a four-part plan to gather and interpret information from schools all over the United States (with some input from schools world wide). The plan includes: a set of initial explorations of promising locations and contacts with teachers, computer coordinators, district superintendents, state level educational decision makers, educational evaluators, teacher trainers, school board people, educational technologists, university and private sector educational researchers, private sector hardware and software providers, and Federal and private foundation supporters of educational technology; a National Conference including the same categories of people; a set of site visits to cutting-edge locations; and production of a set of reports. During the initial phase, a group of representative people from all over the U.S. participated in a Delphi process to obtain a consensus opinion on a large number of concepts related to the study. A second Delphi process will be conducted at the end of the Summer to assist the project staff in digesting all the data and in formulating the final reports.

This NECC presentation will constitute the first public report of project findings to date, including the results of the first Delphi process, the national conference, many site visits, and will include also a preliminary showing of video tape results from site visits.

Presenter

Ludwig Braun
ISTE
Dix Hills, NY

Sponsor: ISTE

NSF Programs in Ethics and Value Studies

Terrel Ward Bynum
Southern Connecticut State University

Rachelle Hollander
The National Science Foundation

Abstract

The National Science Foundation funds research on the social and moral implications of science and technology. Some of the NSF grants fund projects in computer ethics and related subjects. Dr. Rachelle Hollander, Program Director in "Ethics and Value Studies" at NSF, will discuss her own perspectives and goals as Program Director, as well as those of the NSF in general, on research in computer ethics and related areas. Following Dr. Hollander's remarks, two Principle Investigators from NSF-funded projects will briefly describe their research. Time will be reserved for questions from attendees.

Quality of Life and Computer Education

Ronald E. Anderson, Chair
University of Minnesota
Minneapolis, MN

Abstract

This panel will begin with a report on the SIGCAS quality of life project. Then computer educators will discuss selected problems in computing that affect the quality of life of students, staff, and community. These issues include those of ethics, enforcement of ethics, inequities, human values, and risky or unreliable systems. Key questions will be raised, for example, what should students be taught about computing and their implications for the quality of life? What incentives can be established to insure greater attention to quality in computing? Also, how can greater attention be give to the effects of computing on the quality of life of different kinds of people? The audience will be encouraged to raise additional issues pertaining to the quality of life in computing, especially computer education.

Panelists

C. Dianne Martin
The George Washington University
Washington, DC

Barbara Kurshan
Richmond, VA

Richard Rosenberg
University of British Columbia
Vancouver, BC

Sponsor: ACM/SIGCAS

Integrating Technology into the Classroom Teachers...The Important Piece in the Puzzle

Kam Matray
Project Coordinator K8
California Model Technology School Project
Box 1031, Monterey, CA 93942
(408) 899-1434

Abstract

The California State Department of Education (SDE) established the Model Technology Schools Program in 1987 to research and demonstrate effective use of technology for instructional delivery. The SDE funded six sites throughout California. Each site developed its own unique approach for planning and implementing the use of educational technology. The Monterey Model Technology Schools Project is a student-centered approach. They are finding that their degree of success is dependent upon two elements: 1) the design and implementation of a site- or teacher-based planning and decision making model where teachers act as primary change agents. 2) the development of focused plans for technology use in the classrooms authored by participating teachers. These classroom intervention plans include objectives, state framework and school site plan alignment, activities, timelines, budget, and evaluation measures.

Introduction

Educational Technology...much ado concerning this topic today in both government and school forums across the country. Many educators are finding that the successful integration of technology into the way we do our business involves advancing through a series of stages (Mecklenburger 1988), much like putting together a complex jigsaw puzzle comprised of a plethora of pieces. In California, Monterey Peninsula Unified School District's Model Technology Schools Project (MMTS) has found that the degree of success in putting the pieces together depends upon school-based decision making. More specifically it is dependent upon the development and implementation of project-based Classroom Intervention Plans (CIPs) developed by teachers with the help of their principals, the MTS staff, and research team.

California Model Technology Schools Projects

Three years ago the California State Department established the "Model Technology Schools" Program (MTS) to demonstrate the effective use of technology in instructional delivery. The program is based upon a plan with three key elements: developing partnerships with business and industry, conducting research, and disseminating successes. (Barkman, 1988). Funded through the California State Department of Education and Assembly Bills 803 and 1470, grants of $500,000 per year for five years via an application process were awarded to five school complexes in districts across California. (A sixth MTS site was added in Spring, 1988.) The location of these projects are: Alhambra City and High School District, Cupertino/Fremont Union School Districts, Hueneme Elementary Unified, Los Angeles Unified, Monterey Peninsula Unified, and Sacramento City Unified School District. These school complexes, when fully implemented, will serve as centers for research, training, and demonstration of the integration of technology in the K-12 educational environment. They are also to provide policy guidance to government and business decision makers as to cost-effective strategies for meeting California's education goals.

Six major goals are shared by all MTS sites:

1) Develop and utilize instructional strategies which will support and expand the state curriculum frameworks.
2) Utilize technology in all aspects of school operations with emphasis on instruction.
3) Identify uses of technology to develop communication, problem solving, and high order thinking skills across all curriculum areas.
4) Identify ways to incorporate technology into a variety of instructional strategies including collaborative learning.
5) Develop specific district and school level planning strategies for the coordination and integration of technology into existing programs.
6) Develop improved and more varied methods for assessing the impact of technology on instruction.

Commonalty in the purest sense, however, ends here. Each project has developed its own unique approach to meeting these goals.

The Monterey Project

The Monterey Model Technology Schools Project (MMTS) has two fundamental objectives. The first, as stated in its initial project proposal is, "...to make it possible for students to utilize skills and knowledge acquired from their education (content) combined with the tools now provided by technology to make independent, self-directed, self-regulated decisions that will optimize their ability to solve complex problems both in and outside the classroom." (MMTS Proposal,

1987) The approach to the use of technology in education envisioned in the MMTS project is therefore, defined as a "proactive student-centered approach."

The second goal is to design a site- or teacher-based planning and decision making model—a model sometimes referred to by project teachers as "bottom-up." The philosophy of the MTS approach is that teachers need to be making the important decisions. The validity of this approach was echoed in *Power On!*, the 1988 report from the Congressional Office of Technology Assessment where it was stated that: "Given the right circumstances, teachers could choose the appropriate way to reach their students. With the computer and other tools, the range of opportunities increases. But teachers have to be allowed to choose, willing to make choices, and qualified to implement their choices effectively."

With project philosophy and theoretical framework in place, Monterey Peninsula Unified School District was granted funding July 1, 1987. During the first year of project implementation, capitalization took place at the two elementary sites chosen for participation in the MMTS project. Year Two brought on board the middle school and Year Three finds the high school beginning implementation. Year Three also signals beginning activities for dissemination of successful programs and practices at the elementary and middle school sites. Research findings and development of policy recommendations on the impact and appropriate use of technology on teaching, learning, school management, and home/school relationships is under way (Cradler, 1989).

In the Beginning

Year One of project implementation started with a myriad of activities attended by the participating staff at both elementary school sites; this included all classroom teachers, 85% of the site specialists, principals, the Assistant Superintendent of Instructional Services, the Director of Staff Development, and the MTS Staff (the Project Director and coordinators for Research and Evaluation, Curriculum, and Professional Development respectively.) These initial activities fell into one of four categories:

- to operationally define project concepts (e.g., What exactly is a "proactive" learner?)
- to establish student and organization or staff goals.
- to provide an overview of available technologies.
- to plan site decision making models.

Both sites established curriculum committees composed of grade level representatives and a specialist representative to identify, plan, and align project activities with school plans by curriculum area. The activities of these groups were reported to smaller Site Planning Teams comprised of key planners, chosen by the faculties-at-large. The Site Planning Teams operate as a clearing house for proposed project activities and make procedural decisions on a regular basis and direct recommendations to the faculty at large during weekly faculty meetings.

It was during the first year that these two groups identified the pieces of the "technology puzzle" facing them. The pieces were major questions to be addressed and were related to technology use and curriculum/framework alignment, student/school needs, varying levels of concern and use, hardware acquisition and configuration, software preview and selection, staff development, parent relations, instructional strategies, budget implications, impact on other school programs in site plans, research and evaluation considerations, not to mention timelines. Sorting through all these issues required a strategy for initial efforts at each site. Planning and building a site-wide technology use plan was, therefore, the major goal undertaken by the Site Planning Teams. The endeavor was facilitated by the curriculum committees where curriculum-specific goals and objectives were developed for technology use across the grade levels. However, when it came to real-time implementation of this plan by the classroom teacher, in an oft times isolated classroom with 30 students, initial efforts lacked a degree of focus, especially by those teachers who were "self-professed cyberphobes."

It became clear that a process was needed by which individual classroom teachers could decide exactly how they would bring technology use into their classrooms and do so in a manner that was in alignment with state frameworks. A case in point—Year One found the establishment of one very well-planned, comprehensive "school-wide strand" which defined a scope and sequence for keyboarding and word processing at one project site that articulated naturally with the district Language Arts curriculum and state framework. On-going California Writing Project and cooperative learning strategies professional development activities resulted. However, the MMTS Research and Evaluation coordinator reported that the results of the first year qualitative evaluation of awareness versus application by individual teachers suggested a clear need for definition of *specific* expectations for teachers participating in the project above and beyond the identified site-wide strands (Cradler, 1989). The MMTS Research Coordinator's recommendation was that teachers or teams of teachers define, develop, implement, and evaluate specific Classroom Intervention Plans (CIPs) for the use of education technology and/or student-centered teaching strategies that would:

- Integrate technology into an existing curriculum framework area.
- Address the local school improvement plan.
- Support one or more of the MMTS objectives.

- Operationally define the technology and/or the student-centered instructional strategy to be implemented.
- Identify activities to be implemented by support personnel.
- Describe staff development and materials needed.
- Describe anticipated products and/or practices to result from the CIP that could be disseminate.
- Develop with the R&E team, a comprehensive evaluation plan for the CIP.
- Identify CIP development and implementation activities and timelines.
- Determine development and implementation costs for the CIP.

These recommendations were based upon a procedure developed by the Research Coordinator for educational technology planning in South San Francisco Unified School District and the Teacher Education and Computer Center. Using this approach it was found that programs and projects were more effective if staff were involved in a systematic planning and development process whereby staff developed projects around their *own* ideas—ideas which were reflections of school needs. This process was necessary to create the sense of ownership and commitment at the individual teacher level (Cradler, 1987); it was also in line with project's "bottom-up" philosophy.

Putting It All Together

Based upon these recommendations, the MMTS staff adapted the Classroom Intervention Plan model to provide a "handle" on the integration process, provide an immediacy of ownership for the innovation undertaken, and, simply put, to provide a structure and focus. The original draft of the CIP process was *beta tested* by a team of representative teachers, chosen by their peers. After the draft was revised, based upon teacher recommendation, summer CIP development sessions were scheduled for Year Two—also a teacher recommendation. As a result of this process, 29 definitive plans were initially developed by individual teachers or teacher teams at the elementary level to integrate technology into the delivery of the curriculum. By the end of the first semester of Year Two, 21 additional plans were developed at the middle school; 22 more developed at the high school by the same time Year Three. It should be noted that by the time the middle and high schools began their CIP development process, significant revisions had been made to provide a structure for preplanning and to effectively decrease paperwork and management time. Each CIP includes a description of the participating student group, subject matter emphasis, curriculum framework, and school plan objectives addressed. They also include descriptions of objectives and activities with timelines that are planned for both students and teachers. With the assistance of the MTS Staff and Research Coordinator, professional development needs, procedures for evaluation, possible dissemination products, as well as needed materials and resources are identified.

One page of each CIP bears special mention—the budget page. As the CIP planning/writing phase neared completion at each level, it was not surprising that initial capital outlay requests exceeded each site's budget allocation. At one site the principal facilitated a total-faculty meeting addressing this issue. Faculty members split into brainstorming groups to come up with strategies for trimming expenses across all CIPs. The teachers agreed upon three cost cutting measures: limiting the number of substitute days requested per teacher for implementation, delaying purchase of two camcorders until Year Three, and sharing laser players instead of purchasing one per grade level. At the other site, the Site Planning Team found that highest expenditures lay in software allowances. With this in mind, the group made a recommendation to their faculty to keep the substitute days and stipends requested intact, but to standardize software allowance for all project teachers or teacher teams. The recommendation was accepted and, again, the budget balanced. As a result of this process, teachers were not only more careful with software preview and purchase, but also valued and utilized their release time for project planning and implementation all the more. Myron Bursheim, principal of futuristic Oxbow Creek Elementary in Andover, Minnesota, has a message that is all too relevant to this experience: "Empowerment is crucial. In order for [technology in schools] to work, the initiative has to come from the staff. That leadership and ownership among teachers is critical." (Barbour, 1989)

The success of the CIP process, to date, is linked with the clear understanding by participating teachers, staff, and administrators that site-based decision making is "a joint planning and problem solving process that seeks to improve the quality of working life and education. As the name implies, it is *decision making*, not another word for input." (Huts, 1989) There are as many ways to undertake integration of technology as there are educators taking that stride forward—each unique in their own way, often dependent upon teaching style and professional priorities set within a given teaching philosophy. The CIP process, based upon a project philosophy of site-based planning, continues to be the vehicle for focused, incremental change in a manner that honors individuality within the group.

For all the success presently being encountered in the MMTS project's "bottom-up" decision making model, it must be noted that the process is not free of risks for either project participants or staff. There are those teachers who are not comfortable in this role—

- Not everyone knows how to effectively and efficiently make a decision. For that matter, not everyone is able to effectively identify the decision that needs to be made.

  School decision making *takes more time*...in the short run. In the long run, however, if the whole process of decision making *and* implementation is considered, it is the *least* time-honored method.
- The decision makers need to be able to take the responsibility for the outcome of decisions made. (Sometimes this does not happen.)
- Accept the reality that some participants may not want the privilege [responsibility] of decision making.

Some of the unseen benefits, however, are:

- Celebrate the fact that school-based decision making reduces isolation from colleagues and seems to increase teacher creativeness.
- Such a model does not threaten the principals; rather it can enhance the principal as instructional leader and lead to improved learning (Emperors, 1989).

The Future

Year Three of the Monterey project is in progress. The teachers at the high school have self-seeking their first Cadmus of colleagues for participation. Their CIPs have been planned and written; implementation started early on in 1990. The CIP process utilized during Year Three include refinement suggested by Year Two findings. These included increasing attention to operationally defining CIP outcomes for teachers and students with more focus on making the CIP a "working recipe" to be followed when implementing the intervention strategy. (Cradler, 1990) During Year Three the middle school and two elementary schools have also expanded and refined their CIPs toward departmental and grade level plans that are more closely integrated into respective school plans and now are planning for the identification and dissemination of promising practices.

The CIP process lends itself well to change. It is a dynamic process wherein teachers are authors of their own innovation and, as a result, can effectively and efficiently make the changes on an as basis. Too, it realizes that change is a process, not an event. The project evaluation for Year Two showed that all participating teachers viewed the process as a major contribution to their own effectiveness in using technology to support and expand class instruction (Cradler 1989). Within the MTS project the CIP process effectively linked with site-based planning has confirmed that the integration of technology into the delivery of the curriculum must be related to people first and the innovation second (Hall, 1978). The CIP protects the integrity of individual style, but maintains accountability of the system as change takes place.

Literature Cited

Barbour, Andrew. "Teachers: The Real Key to Solving Curriculum Problems." *Electronic Learning Special Supplement*. October, 1989.

Barkman, Ira. "Model Technology Schools." *California Model Technology Schools* 1,1. January, 1988.

Cradler, John. *Policy Recommendation for Program Improvement with Educational Technology in California Schools*. Policy Analysis for California Education (PACE), Berkeley, California. 1987.

Cradler, John. *Monterey Model Technology Schools—1988-89 Research and Evaluation Report*. California State Department of Education. July, 1989.

Cradler, John. *Monterey Model Technology Schools—1989-90 Mid—Year Research and Evaluation Report*. California State Department of Education. March, 1990.

Empey, Donald. "The School Principal—An Endangered Species?" *Thrust* 19,1. September, 1989.

Hall, Gene and S.F. Loucks. "Teacher Concerns as a Basis for Facilitating and Personalizing Staff Development." *Teachers' College Record*. 1978,80.

Hutto, Tommye. "Exploring Site-based Decision Making." *California Teachers' Association Action* 28, 2. October, 1989.

Mecklenburger, James A. "Bold Pacesetters are Transforming Schools for New Times." *The American School Board Journal* 175, 9. September, 1988.

The Monterey Model. Today's Educators Using Technology to Prepare Students for the Future. Monterey Peninsula Unified School District: A proposal to the California State Department of Education, Office of Educational Technology, Sacramento. 1987.

Power On! New Tools for Teaching and Learning: A Summary. Office of Technology Assessment SET-380. Congress of the United States. September, 1988.

Computing Across the Curriculum: An Integral Part of Preservice Teacher Education

Nancy Cunniff
St. Bonaventure University, School of Education
Box 161, St. Bonaventure, NY 14778
716-375-2363 BITNET: Nancy@Rsage

Abstract

This paper discusses the urgent need for broadening the view of computing in preservice teacher education. Single courses in computing and its use are an important, even crucial aspect of the program, and discussion of computer use in methods courses is critical. However, unless teacher education students experience computing *as a regular element of the general instructional program*, it is unlikely that they will consider computers to be as general and as flexible a teaching tool as they perceive chalkboards and textbooks to be. This paper presents some concrete examples of computer use in a general Introduction to Education course. The suggested activities can be integrated into a course using only one or two computers, thus not requiring extensive facilities. The paper strongly suggests that the critical component for improving computer use in education is the development of College and University faculty as fluent users of technology both personally and in their teaching.

Introduction

Education in America is stubbornly resistant to change. During the past few months, I have had the chance to look at classrooms in many elementary schools. At first glance, classrooms look pretty much the same as they have for the last 10, 20 or 30 years. Despite the vast changes that have taken place in our world, students in many schools continue to sit in desks in rows, listening to the teacher talk, reading "round-robin" from textbooks and writing spelling words "ten times each." On closer inspection, however, one notices a computer in many classrooms. Often it is hidden in a corner or piled high with workbooks and dittos—but it's there. Sadly, over the course of several months, only once did I see a classroom computer actually turned on, only once were children engaged in learning using the computer. That was in a fourth grade classroom where a group of children were excitedly using CAI programs to practice facts. I only once observed a teacher using a computer—running a grade-book program.

Oh, yes, children are using computers in lab settings. Classes typically go *en masse* to the computer lab. Since "Computers" is treated as a "Special" in most schools, the computer teacher works with the class while the classroom teacher hurriedly disappears for a break or planning period. Even where exciting things are happening in labs, there is little coordination with what goes on during the rest of the day. Yes, computers are in many schools, but no, they don't seem to be making inroads into the general fabric of the teaching-learning experience in regular classrooms in any significant way. Computing is not an integral part of elementary education.

If computing will become an integral part of education, then I propose that it must be an integral part of teacher education. This is not a new idea. Many articles and course proposals have been written, and seminars and conference sessions have been held—the NECC 1988 and 1989 conferences were host to many such discussions. Most of those, however, focused on teaching computing in the preservice program: how and what to teach, whether there could be consensus about the design of a single course, and so on. The project sponsored by the ACM in conjunction with Teachers College provided an in depth look at these issues (SIGCUE Outlook, Fall, 1988).

It seems that many preservice students are taking a course in computing. *Power On!* the OTA report released in 1988 indicates that most Schools of Education have some course offerings in computing for their students (Office of Technology Assessment, 1988, p. 209). However, it seems that too few students see or experience enough computing to have any real, long-term effect on their teaching. One isolated course, while better than nothing, does not a computer-using educator make! The fact is, that even with a supposedly large number of educational computing courses being offered in colleges, "teachers are emerging from their preservice training to become part of the problem of integrating technology into the classroom rather than part of the solution" (Lawson, 1988, p.1).

Simplistic as it may sound, teachers *will* continue to teach as they were taught. If we are to change the traditional classroom, we have to begin to do it on the University level. We must teach our preservice students the way we want them to teach. We must depart from the traditional university lecture approach and teach using computing, modeling a wide variety of uses of computing. We must first understand, believe and espouse the fact that *without effective preservice education computing will not be used effectively in schools any time soon.*

This paper examines some of the problems facing Schools of Education and describes some examples of integration of computing into a general, non-computing course. The focus is to involve computing in the general courses for preservice students so that they begin to see its application.

Computing for Preservice Students

Background

Computing is not a regular part of the college experience for most students. In an informal survey of the small group of college freshmen I teach (all of whom plan to major in Education), I found that most did not have any regular computing experiences in high school. Of those who did have a course in computing, most expressed negative reactions, and suggested that what they "learned" was either irrelevant or just plain boring. Further, in a survey of 60 Elementary Education majors (21 seniors and 39 juniors), only 14 considered themselves "competent computer users." Even fewer currently use a computer regularly, even for word processing. Of the students surveyed, only 11 used computing in high school and only 13 responded that computing was a part of their college experience thus far. It bears repeating that the students surveyed are all majoring in elementary education and will be in the job market or responsible for teaching in America's classrooms within the next two years.

During the past few years, there has been considerable emphasis on improving the quality and increasing the availability of preservice computer education. Projects focusing on coursework in computing for preservice students are not enough. If we expect teachers to use computing *as a regular part of their teaching repertoire*, then the preservice educational experiences of students studying education must be replete with computing.

I suggest that one of our failures on the University level is that too often we *talk about* and *teach about*, but we do not *use*—in short, we do not teach the way we are telling our students to teach. In a discussion of the limited impact of technology on higher education, McNeil emphasizes that, "we have not even come close yet to making a difference in the lives and learning experiences of most Americans. Millions of people have not and will not touch a computer" (McNeil, 1989, p. A44). The reality is that in college and university classrooms, there is virtually no integration of technology in the regular, day-to-day teaching. Traditional lecture formats still reign. Students continue to listen while professors continue to talk -- even in courses in computing!

Professors in schools of education are as guilty—or more guilty—as those in the rest of the college. Most faculty members would agree that we should do something about the limited uses of technology in education classes. At the same time, most faculty members quietly admit that they don't really know how to use a computer—and they have no idea about how to integrate computing into their methods courses. And so, our graduating students may have heard about computing in education, they may even have had a course in educational computing, but they have not seen computing used as a teaching tool.

Although only a few states require newly certified elementary teachers to have pre-service coursework in computing and/or technology, it seems to be widely accepted that if we are going to realize the potential of computing in education, preservice teachers must experience computing as a regular part of their instructional program. Unless computing is *used* as part of the mainstream program of the teacher-education student, it is not likely to be used in any significant way by the students themselves. They will probably not see computing as a priority, and thus will not make any attempt to incorporate its use into field experiences and student teaching. Therefore, it is likely that such students, even if they have had coursework in computing, will *still* graduate without any significant experience in computing as an educational tool.

"Whole Computing"

Taking liberty with the Whole Language movement, I suggest that the crux of the problem is a failure to view computing as whole. We suffer from the segregation of skills—much as language arts suffers from the segregation into handwriting, syntax, phonics, spelling—forgetting that the big picture is *language*—communication, reading and writing. In computer education we seem to have fallen into the same trap. We teach keyboarding and the skills of word processing. We teach programming syntax but not design. We fail to use the skills in an integrated fashion.

The solution, of course, is for Education Faculty to become more fluent users and make a firm commitment to integration—in computing as in all other areas.

A Proposal: Use Computing in General Courses

In his explanation of the ACM project on Computing in Preservice Education, Robert Taylor set forth some training maxims. His fifth maxim, "Capitalize on the opportunity technology presents to make changes where appropriate in the curriculum and teaching methods" (Taylor, 1988, p. 5), sets the stage for the solution I propose here.

The Problem

Talking about computing and computer use rather than modelling the effective and appropriate use of computing bypasses the issue of how to use computing effectively as part of the regular instruction program. Without experience using computing and seeing the logistics of organizing classes for computer use (*e.g.*, how to deal with a whole class and one or two computers), future teachers will only have a textbook knowledge of how to use computing. The current buzz word in education is *INTEGRATION*. We talk about it, we argue for it, we even grade our student teachers on it—but we are not practicing it ourselves. "The vast majority of faculty members and administrators simply have no sense of the implications or the possibilities of using technology to teach, counsel, and administer. They either ignore technology or stubbornly resist it" (McNeil, 1989, p. A44). Faculty members in most disciplines, including those in Education, have little experience with computing, little interest in it, and a great deal of fear and anxiety about it.

A Solution

All courses in the education department should be taught *using technology as a standard pedagogical tool* when appropriate. In other words, computing should not be relegated to a single course, or even to the methods courses; its use should be regularly and effectively modelled by the faculty in all courses so that students, in the course of learning about education see that technology can be used to teach.

I teach several courses in the School of Education, among them *Methods of Teaching Language Arts* and *Introduction to Education*. Both of these courses are required for students majoring in Elementary Education. Like many other preservice programs in education, we are a technology-poor department. Although there are plans for a new computer lab, until very recently the department had a single APPLE IIe computer. We now have (on loan) three Apple IIGS computers. While there are some other computer labs on campus, they are usually fully scheduled by other departments who have priority. I was determined to figure out a way to include computing in my general education classes using extremely limited resources.

In preparing to teach the courses for the first time, I began to focus on using computers in my teaching, not teach about computing. I recognized and attempted to deal with three major questions:

How can I organize the content and present it so that the students approach it with curiosity and intellectual enthusiasm?

How can I incorporate computing so that I can fill a gaping hole in their backgrounds (there is no computing currently being taught in this School of Education) and begin to give these students an appreciation of the use of technology as a teaching tool?

How, in an environment that is technology poor, can I model the use of computing so that the students see that one computer can be used effectively with a large group and that small groups of students can work cooperatively and successfully using one computer?

Examples from Introduction: A Traditional Lecture Class

For most of the students *Introduction* is the first course in the major. The course covers such things as the history and philosophy of education, governance and fiscal control of schools, social factors affecting schools, and an introduction to curriculum development and a (very brief) look at teaching and learning styles. The students see this course merely as a requirement, they are not terribly excited about taking it. They see the material as dry and certainly do not see its relevance.

1. History of Education

The students were assigned to work in groups to examine the textbook material about the history of education. One semester, each group focused on a different topic and created a timeline. One group worked on the most influential figures in educational history from Socrates to the present. Another traced time periods: The Greeks, the Romans, the Middle Ages, and so on. A third group traced those figures influential in the development of scientific thinking, while a fourth traced the development of the effect of religion on the historical development of education. A second semester, the students were assigned the reading in their text, were divided into small groups (approximately six in a group) and prepared a timeline of what the group perceived as the important points of the readings.

Using Tom Snyder's *Timeliner*, each group entered their timeline. This software is so easy to use that little introduction was necessary. We began as a whole group projecting the program using the overhead plate. I demonstrated how to start the program, showed the critical keystrokes (of which there are very few), and answered questions. Each group entered their own timeline, and we printed and reproduced copies for the class. The following class we examined the timelines together, noticed trends and similarities in the ebb and flow of educational change. We discussed the readings further and came to some consensus about the important points. The timelines were then used as study guides for exams.

2. Philosophy of Education

Students study educational philosophy to get a sense of why we do what we do, but the importance and relevance often seems to be missed by beginning students. In an attempt to help the students see that the ideas of philosophers across centuries are interconnected, we worked with a database of key ideas. Together, the students and I developed a database template—although I did not tell them at first what we were doing. We determined what were the key pieces of information that they should know about each of the philosophers in question. I guided them to include a place for "key words about the philosophy" so that we could later use this as a search field to make comparisons and connections. In pairs, students researched two of the key philosophers, filling out a paper copy of the template.

Using *Bank Street Filer*, we created a database. We used a single computer so that we'd have a single file to work with at the end of the project. Students entered the information during class group meetings. Those students who did not get to enter their data, could use one of the computers in an office accessible during the day. Once all of the information was entered, we used the database, along with an overhead projection plate to run searches and trace trends of thought.

As a final activity, we compared some of the timelines and the database information to get a clear picture of the history of educational thought. This would have been a more successful activity if there were more computers available. The experience provided a good way for the students to organize their ideas, but the time necessary became a problem. I would recommend some possible modifications:

1) use a program which allows database files to be merged so that you can have students work on several computers;
2) enter the database information yourself and work with the created database during class to do searches.

3. Reading Tables and Statistics

There are a lot of tables and charts in most textbooks. Lots of statistics are presented—and most are poorly discussed or explained. If you carefully watch students reading an article or a textbook, you can often tell when they approach a table: the eyes glaze over and the page flips far too soon to suggest comprehension. Most students just do not know how to read tables or make sense out of statistics. A demonstration of table and chart construction using a spreadsheet helps students to understand statistics by giving them a visual representation, letting them see how charts are dependent on the information in tables.

I used an MS-DOS computer (I resorted to bringing my own computer from home) to teach a lesson on "Making Sense of Statistics." Using MicroSoft *Excel*, I copied several of the tables that were in the textbook. With the class watching, I rearranged the order of the tables, quickly made some charts from the information and talked about the need to determine what it is you want to know from a table. We hid some of the columns so that we could more easily access certain pieces of information. We talked about appropriate and inappropriate charting techniques for different types of data.

While this is not a statistics course, and technically, reading statistics is not a part of the course material, it is evident to most of us that students just "skip over" the tables and statistics they encounter when reading textbooks. By providing an alternative representation for the material, the students seemed to grasp the information more accurately.

4. Models of Teaching

One strand of the course deals with what we teach in schools, how that changes over time, and how we teach it. To illustrate different approaches for teaching the same content, several software packages were demonstrated. We discussed two content areas (writing and math). After discussing their recollections of how they learned writing and their current personal reactions to writing assignments, we used several pieces of software in a series of "mock" writing lessons. I began with *Print Shop*—it was motivating and easy to demonstrate. We created personalized letterheads for a couple of students. We then wrote and illustrated a group story using *Kidwriter*. Finally, we entered one student's term paper outline using the *Appleworks* word processor.

After these experiences, I set up small group meetings to teach students how to use a word processor so that they could use it for their required term paper.

Similarly, we looked at several pieces of math software, comparing the philosophy of using CAI programs such as *Alien Addition* to that of using Logo or a problem solving program such as *GNEE or Not GNEE*. An ensuing discussion focused on connecting educational philosophies with the more practical aspects of classroom activities.

Examples from Methods of Language Arts

All education majors take several methods courses. The focus is not the teaching of subject matter itself, but the demonstration, discussion and practice of a variety of teaching and learning techniques. Since there are so many appropriate uses of computing in the Language Arts area, I felt it was a prime area for integration. I expected the students in this course to master some aspects of computing, and so I offered supplemental seminars in word processing. Since word processing is such an important aspect of the language arts classroom, all of the students were required to complete two writing assignments using *Appleworks*. In addition, I used computing throughout the course for demonstration and actual teaching.

1. Production of a Journal

One of the course requirements is to get to know and use professional sources such as magazines, journals and trade books for teaching ideas. To focus the students' work, they were required to write a short article describing some teaching strategy, management technique or classroom activity. The articles had to be typed using *Appleworks*, and at the end of the semester, a group of students (volunteering for extra credit!) got together, learned to use a desktop publishing program (*Publish It!*) and produced an in-house Journal. The first issue was printed in December, 1989 and the second is about ready to be compiled. It has been printed and is being sold by the student Education Association.

The students who volunteered to work on this project were beginners. Before learning to desktop publish, most only had a couple of hours of experience using *Appleworks*. They were so enthusiastic about the project that they spent hours working at formatting, inserting graphics, designing, and so on. The result was excellent enthusiasm. This semester, more than half of the students in the class want to work on the publishing project.

2. Writing Poetry

One of the topics covered in the class is poetry writing. Before the class, I prepared several program in Logo that help create poems. Besides the familiar programs that generate random lines of poetry, I wrote procedures to guide the student in the production of Cinquains and Diamante.

During this class, I did not stress the computer at all—I merely used it as we created class poems. The room was dim so all could see the monitor—and in some ways, this helped the students feel less intimidated about participating in what is often a difficult writing experience. As a result of this experience, many students have asked to use the computer program that we used for poetry.

3. Teacher Utilities

For one of the regularly scheduled quizzes, I created a crossword puzzle using *Crossword Magic*. I mentioned that I had used the computer. Soon there was an fairly constant stream of students requesting to use the crossword program. In fact, several other faculty members took note of this particular activity and have since used it for other classes.

During a class discussion on how to design activities for learning centers, I demonstrated how to make word search and word jumble puzzles using *Teacher's Toolbox*. Again, the focus was on how to customize materials for specific learning environments. The point was that while you can construct such puzzles by hand, it is faster to do it using a tool specifically designed for the task. Again, the number of students beginning to use computers increased rapidly.

4. The Writing Process

Of course, one of the major foci in a Language Arts Methods course is the writing process. College students need to learn this for their own work, but preservice teachers must master the process since it is so much a part of writing instruction at the elementary level. Among writing instructors there is still some controversy about the role the computer should play in the writing process; however, I believe that a process involving free writing, revising, editing and publishing ought to involve computer use. This, above all, illustrates the power and flexibility of technology in the language arts area.

These activities are not always easy to organize. They require a lot of organization (if only how to get the computer to the lecture hall and arrange so that all can see) and a fair amount of extra preparation time. However, they begin to address what education is all about—active learning in an environment of inquiry. It should be stressed that all of the activities suggested require a minimum of hardware. One or two computers and a large monitor or projection plate will suffice. Further, the software requirements are flexible. Any good integrated package will do, but one can use stand-alone versions of the tools as suggested above. Software used for demonstration can be selected according to the needs and interests of the class—and according to local availability. The idea here is to use computing to meet your teaching and learning needs—not to teach about computing!

Conclusion

In a recent popular discussion of educational computing, one common scenario is discussed:

"The 'computer lab down the hall' in which kids use PCs to learn programming or word processing devoid of any broader educational goals has become a sad and all-too-real stereotype for the unimaginative ways that many schools employ computers." (O'Malley, 1989, p. 117).

This is an unfortunate and, as O'Malley puts it, all-too-real statement of the way things are in a lot of schools, both elementary and secondary. But it's even more real in the schools of education where our teachers are being trained. Is it any wonder that computers in the elementary and secondary schools are used in a fashion almost completely disjointed from the rest of the curriculum when what little training teachers receive is in a setting where the focus is computing and not the use of computing to teach? Courses on computing should not be discontinued and, in fact, should be extended. But a strong push to get computing and technology used in general classes must be the next wave of reform in colleges and universities.

References

Lawson, J. (1988). Outlook preservice project issue. *SIGCUE Outlook, ACM*, *20*:1, 1-3.

McNeil, D.R. (1989). Technology is a hot topic, but its impact on higher education has been minimal. *The Chronicle of Higher Education*. June 7, 1989, A44.

O'Malley, C. (1989). The revolution is yet to come. *Personal Computing*, *13*:10, 115-120.

Office of Technology Assessment. (1988) *Power On!* Washington, DC: U.S. Government Printing Office.

Taylor, R. (1988). Outlook: Computing in Education. A single course for preservice teachers. *SIGCUE Outlook, ACM*, *20*:1, 3-5.

Teacher Support for Technology Integration: What Worked; What Didn't

Linda G. Polin
Pepperdine University
Graduate School of Education and Psychology
400 Corporate Pointe
Culver City, CA 90230-7627
(213) 568-5641 BitNet: LPOLIN@PEPVAX

Abstract

This paper reports findings from the second year of a five year, state-funded project to develop and sustain a "model technology school" complex (MTS) covering grades K through 12. Five school districts were approved for MTS funding in July of 1988; this paper describes the progress of the teaching staff in one elementary in one of the five districts. The paper describes a number of formal and informal mechanisms for teacher change used by project staff at the elementary school. Each method is discussed in terms of its relative and differential success. [Note. All school and personnel names are pseudonyms.]

Introduction

Before the state-funded Model Technology Schools project (MTS) came to Loma Vista Elementary School, the technology opportunities for students and staff were few. The K-8 school of 550 students had an Apple//e lab of about twelve computers. The lab was only opened and run by an itinerant computer coordinator from the district who visited the site weekly. Teachers who chose to bring their classes to lab relied upon the lab coordinator to do the instruction. In this setting students were introduced to the graphics side of Logo. In their proposal for funding the MTS project staff proposed a "student-centered" approach to "technology intergration" in classrooms, and they proposed to "saturate" or enrich the technology environment to allow for free and easy access to technology as a tool for learning.

By the end of the first year, every teacher at Loma Vista had two computers and a printer in his or her classroom, and an Apple IIGS at home. The school hallways contained five computer "banks" of Apple IIGS machines. The computer lab had been moved to the larger basement room and now housed approximately twenty Apple IIGS machines with strategically placed printers. Additionally, the school had three camcorders, VCR's and large screen monitors, a tape-editing deck, a satellite dish, and a photocopying machine strictly for students' use and one for teachers' use. A number of software and multimedia packages were available in lab pack or site license quantities including among others: FrEdWriter, LogoWriter (English and Spanish), Lego TC Logo, and the Voyage of the Mimi package, and access to the Kids' Network project. Despite all these marvelous new opportunities, the teaching staff remained largely the same staff that had previously relied upon an occasional visit to the computer lab, led by a district computer coordinator. Clearly staff development would be a main focus of the first two years of the project.

MTS project staff tried several very different approaches to create and support changes in teachers' use of technology in instruction. Some strategies were deliberate; some were unintentional. Some worked well for everyone; others were differentially effective. A few were downright unsuccessful. They are discussed in this section in terms of their effectiveness. They are: 1) summer training, 2) informal systems for individualized support, 3) communication of information and opportunities, 4) administrative mandate, and 5) rewards and incentives in the environment.

Summer Training

Changes in summer training for Loma Vista staff from Year One to Year Two of the project reflect insights of the project staff about teacher needs and their reactions to preliminary research results from Year One, shared by the researchers. In Year One the summer training had the dual foci of communicating project expectations and introducing teachers to technology applications and choices. Teachers primarily observed demonstrations of a variety of technologies, e.g., telecommunications, desktop publishing, Lego Logo, and instructional television. Very little time was spent explaining the "student-centeredness" aspects of the project. Technology demonstrations were more generically tied to instruction rather than specific to grade level curriculum. Outside of brief opportunities in the demonstration sessions, teachers' "hands-on" time was directed toward practice with the Apple IIGS teachers would be taking home with them at the end of the training.

By January of the first year of implementation, project staff were expressing dissatisfactions with the outcomes of their summer training efforts. In May, when the curriculum and training support staff sat down to plan for the Year Two summer training, their planning talk revolved around conveying the "student-centeredness" approach and getting teachers to make instructional plans and commit to implementing them.

These two topics also largely characterized staff and teacher talk in the first year about the missing elements of the first summer's training.

Year Two training ran during a Friday, Monday, and Tuesday in late June, 1988. The sessions opened with a series of presentations by the MTS project coordinator, curriculum and training staff, and the school principal. All three stressed the need for teachers to "deliver" in Year Two. On the handout describing the three-day training teachers were told, "In three days you will have a direct outcome, a planned project with it's (sic) framework/curriculum tie-in, materials needed, goals, list of participants, expected outcomes. etc." Teachers were further asked to include in their plans the approximate date they would implement their project during the upcoming school year.

In planning the summer training sessions and during the sessions themselves, MTS curriculum and training staff talked about the need for teachers to adopt a cross-curricular "project" approach to instruction. Their talk about projects coincided with their talk about "student-centeredness" in ways that suggested they saw the "project approach" as a way to accommodate student-centered instruction. Up to this point, student-centeredness had largely been defined as "students making their own choices" about instructional tasks. In the summer training, the MTS staff explained and modeled project development with teachers making decisions about what would probably interest students and where in the sequence of instructional events students could "take responsibility" and "make choices." A lot of those modeled opportunities arose in "cooperative learning" groups. Though no summer training time was spent discussing cooperative learning, the school district had continued to offer in-service workshops during the school year (for at least the last two years) on cooperative learning in the hopes of having all teachers eventually trained in cooperative learning strategies. (Many of the Loma Vista teachers had been through this workshop, and a few needing it took it during Year One.)

The first day was organized around modeling the project planning process, as carried out by two MTS staffers posing as cooperating teachers. Following the model, teachers worked in groups to "brainstorm" topics appropriate to their own curriculum and grade levels, and that would be good candidates for the student-centered approach. The modeled project was a cross-curricular simulation. As teachers watched, the modeling pair consulted state and district curriculum guidelines, made notes about the resources and support they would need, and the ways in which they would get students to "make choices and take responsibility for their learning." The following two days allowed teachers planning time, with available curricular materials and support staff. Teachers were encouraged to work with a teacher buddy to plan a project involving both classes or simply to use one another for support. Teachers were given a handout to use as a "self check-list" for their project planning.

The last day of summer training came at the start of the second school year. Early in September, before students returned, teachers met on site. At that time they asked many questions about cooperative learning. The MTS staff member who conducted the meeting admitted a limited knowledge of cooperative learning and called upon those in the room who had attended district workshops to respond to colleagues questions. Most questions were procedural and logistical: "how do you place kids in groups?" "What if kids don't work well together?" "How do you give grades if someone hasn't done much work?" Teachers talked mostly about the need to teach their students how to be in cooperative learning groups. The differences between cooperative learning groups and "generic" small group work was not discussed.

Evidence of Impact from Summer Training

Dissatisfactions with the effectiveness of the summer training preceding Year One led to dramatic changes in Year Two summer training. Primarily staff trainers shifted the emphasis from exploring technology to planing for teaching. Teachers were strongly reminded of the need to align instruction with state and district frameworks by using them to guide instructional planning. Teachers were given "checklists" to use in monitoring and revising their plans. Teachers were given time to use in planning for a new "project" oriented approach to instruction.

During the school year very few of the projects teachers planned during summer training were carried out in their classrooms. This was partly because the projects were planned at a very gross level and were not filled in with detail. As a result they tended to be too encompassing and difficult to "get a handle on." Although teachers did make use of the district and state frameworks in planning their projects, they clearly had difficulty believing they would "cover" their curricular obligations by carrying out the grand projects they had proposed during the summer.

While the summer training did not meet its stated goals of making teachers plan student-centered, technology-rich projects to carry out during the school year, it did succeed in moving teachers further along in their ability and willingness to eventually do just that. First of all, the term "project" became salient to teachers as the object of their training and the goal for them for the second year. They talked about "projects" when they talked to each other, to students, and to project staff. They began to formulate their own definitions of "project" and check them out with MTS staff, often asking "Would this be a good project?" and "Would that count as a project?"

The other consequence of the summer training was the clear sanctioning of teacher teamwork and peer support. Teachers had been told to work in groups either by grade level or by project interests. This aspect of summer training emphases was the most readily identifiable during the school year. Many teacher pairs (both cross grade and within grade levels) worked on instructional projects during Year Two. This was not true in Year One in which teachers were mostly teamed with support staff rather than each other. Thus, in a sense, this shift toward teacher teaming represented a kind of "weaning" from staff support, and it came at a good time, i.e., when the availability of on-site support was significantly reduced. Were the costs associated with the summer training worth the consequences? Probably so for the MTS project. As described below, there was a great deal of "movement" toward the dual goals of student-centeredness and technology integration in Year Two of the project. And yet, on-site staff support diminished in Year Two when some of the project staff moved to the high school to assist with the start-up of the MTS project there. While some momentum from the experiences of Year One is probably a strong contributor to the changes seen in Year Two, it is important to note that the kind of changes evident in teachers more closely paralleled the emphases in summer training than the emphases and experiences of Year One.

What didn't work in the summer training? Clearly things could have been improved, and no doubt the MTS staff will act on the shortcomings they perceived in the training sessions. However, one element may continue to be overlooked and so it is raised here. In both Year One and Year Two training, the teachers expressed a concern that their instructional aides weren't being included in the training. In Year Two, aides were invited to attend, however, they were treated as a separate group and considerably less training time was spent with them. In their meetings they were generally led to "brainstorm" about their needs or about their concerns. They did not plan with the teachers when the teachers worked on project development, despite the fact that these very aides have a significant role in the enactment of instructional plans. The teachers, as well as the aides, expressed dissatisfaction about the summer training arrangement for the classroom aides. It is important to recognize the implication of this problem for subgroups of students who are limited-English proficient (LEP) and find themselves in classrooms with monolingual English-speaking teachers. In those classrooms instructional aides play a significant instructional role. To the extent that the MTS project seeks to involve all students, it must involve all staff with an instructional role.

Informal Systems for Staff Development and Support: Models of Teacher Support[1]

In Year One the MTS staff tried two approaches with mixed success: SIG's and Buddy System. Growth in teacher expertise and application of MTS concepts was limited where it relied upon the SIG system, with a few exceptions. On the other hand, the "Buddy System" met with enough success to ensure its continued use in Year Two, although in a much "altered" state.

Teacher Special Interest Groups (SIG's)

The SIG (special interest group) was initially conceived as a natural follow on from the teachers' self-evaluation of interests and skills in a variety of technologies. Thus, SIG's were formed on topics such as telecommunications, interactive laserdisc, or camcorder. Each SIG was headed by an MTS staff member who called after school meetings to "train," answer questions, and assist in instructional planning. The SIG movement was abandoned as it began to fall apart because of the inability of SIG members to all meet at the same time, or at a time convenient for the MTS staffers. (See Year One Final Report for detailed descriptions.[2]) In place of the SIG's, staff members divided up responsibility for the faculty amongst themselves, taking responsibility for approaching individuals to assist them with instructional planning for technology use. This plan was more or less successful depending upon the match between personalities and support styles of the MTS staffer and the teacher she was responsible for. Some teachers were virtually left alone or "abandoned" as "hopeless" or not worth the time and energy, often after planning sessions between the two did not come to fruition, or after MTS staffers perceived themselves to be "stonewalled" by teachers who repeatedly cancelled appointments with them.

Buddy System for Teacher Support

In the Spring of Year One MTS staff recognized the limited effectiveness of the SIG approach. In its place they initiated the "buddy system," which linked each support staff member with a number of teachers for whom he or she was personally responsible. At that time all MTS staff viewed these efforts primarily as ways of dividing up the responsibility for supporting eighteen teachers among four on-site staff. With the goal of "coverage" in mind, the groupings tended to emphasize individual teachers' interests and skills rather than teachers' natural partnerships by grade level or shared values, as was the case when teachers grouped themselves during the Year Two summer training.

At the beginning of Year Two the "buddy system" was still in operation, though with the loss of one support staffer to the high school, resources for this approach were strained. Soon each of the remaining MTS staff members settled into a particular kind of support style, described below, and particular teachers gravitated toward particular staffers. As described above, the presence of the teacher teaming, in planning and implementation, was a new development in Year Two, and seemed to have a positive effect.

---

[1] A preliminary version of the analyses that form the basis for this section was presented at the Annual Meeting of the American Educational Research Association, San Francisco, March, 1989.

[2] The First Year Final Report may be obtained by writing Dr. Polin, Pepperdine University-GSEP, 400 Corporate Pointe, Culver City, CA 90230-7627.

Observations of and interviews with MTS team members and school teachers revealed sharp distinctions in the nature of "support" available to teachers by their buddies on the training and curriculum support team. Four kinds of support described here are called: a) directive, b) consultative, c) collaborative, and d) reflective. The variations among categories arise along several dimensions: a) source or initiator of contact; b) party primarily responsible for success or failure; c) relationship to teacher; d) source of instructional focus; e) product structure; f) content of transaction between specialist and teacher. These dimensions defined the nature of the relationship between the teacher and specialist. These patterns were first hunches in Year One, and later confirmed in Year Two data collection and analyses.

Directive (Proactive)

The specialists who fall into this category viewed their job as creating and disseminating lessons or activities. They approached teachers directly, without being solicited. They gave teachers instructional materials, such as dittoes, diskettes, and lesson plans. They often taught a class lesson or led an activity for a teacher, with the teacher present but not participating. They did not follow-up on lessons or activities left with the teacher, and rarely did the teachers working with these specialists seek them out when an activity went awry. Directive staff tended to develop and disseminate activities and lessons that focused on lower level objectives and discrete skills. In the area of technology use, the mastery of the equipment or knowledge of how to use the software was often the objective. Students learned LogoWriter to learn LogoWriter. The source of instructional focus was the curriculum, limited only by the expertise of the specialist. They talked about "transferrable" instructional lessons that could be used elsewhere in the school or district. Teachers main complaint was about the quality of the lessons when lessons didn't work as planned. However, for the most part teachers working with directive staff were pleased with the help they received. These teachers tended to talk about the activities and lessons developed and led by the support staff as their own. This seemed to give them a "right" to feel that they were implementing the MTS project. In Year One, these teachers often had technology activities going before other teachers did, although clearly they depended upon the support staff for those activities.

Consultative (Reactive)

The specialists who fell into this category viewed their job as solving problems and giving advice, i.e., consulting. They did not approach teachers, but rather waited for teachers to come asking for help. Most of their activity was directed toward obtaining and distributing materials when requested, finding out information in response to a question, and solving technical problems (or referring problems to district level specialists, e.g., to repair disk drives). They did not teach classes with or for teachers. They did not offer fully developed ideas or lesson plans, though they might suggest ordering or using a particular videotape or software package. These specialists assumed a very powerful role as "gate-keeper" of knowledge and resources. When given a problem to solve, these specialists solved it and returned with the answer, rather than involve the teacher in the solution process. Teachers main complaint about this sort of support arose when materials and solutions were not received in a timely manner. For the most part teachers receiving this sort of support were quite pleased with it, though the support staff operating in this manner tended to report teachers needled them or became "too demanding."

Collaborative (Negotiated)

The specialists who fell into this category usually approached teachers formally (by arranging appointments) or informally but directly (during social settings such as recess break in the staff room). On some occasions they might also begin their collaboration when approached first by a teacher. Collaborative support staff asked questions about what the teacher was currently doing in class and generated ideas based upon the teacher's response. Those ideas might be rejected or modified by the teacher. They did not teach for or with the teacher, but they did visit classrooms occasionally to chat about or watch the planned lesson. They always knew when the teacher was scheduled to do the planned unit and always followed-up during or after implementation by asking questions. They used their own expertise about teaching and technology to guide and focus the teacher's decisions. This meant they did not always choose to accept and extend every idea teachers came up with, and often spent considerable time and talk to "help the teacher see" the problems with what they considered to be inappropriate choices and plans. Teachers' main complaint with this approach was the amount of time required to generate a single unit of instruction. Though, without exception, teachers supported in this manner carried out instructional plans they were clearly proud of and considered largely successful.

Reflective (Inquisitive)

No staff support specialists on the project fell into this category. This category was derived from analysis of interactions between the researcher and the teachers involved in the case studies. These teachers made remarks on a number of occasions to a variety of audiences that they "could not have done it without [the researcher]." The researcher offered no help, provided no materials. When the teachers who made these remarks were asked about their comments they reported that "simply by being there" and asking questions the researcher was "showing an interest" and provided "moral support" during what teachers considered to be risky instructional moments. And, that the researcher's questions caused these teachers to "think about" what they were doing and why. Teachers' main complaint with this approach was the lack of evaluative feedback on their efforts. The researcher in this category observed in classrooms intermittently throughout an instructional unit (e.g., anywhere between two and twelve weeks). In addition to noting observations, the researcher informally discussed teachers' instructional processes, plans, decisions, interactions with students by asking questions such as: why did you do this; how did you know that; what will happen next; where did you get the idea to do that; tell me about him or her. The researcher shared field notes with teachers on their request (which was rare); passed information between teachers often inadvertently; and asked instructional questions during social occasions (e.g., in the staff room during recess). Interactions with teachers were non-evaluative and non-directive; and teachers rarely asked the researcher questions or advice. On the other hand, they frequently approached the researcher in the hall or elsewhere to report on anecdotes about their classroom which reflected things they were proud about, surprised by, amused by, or frustrated with. These teachers seemed to be the most completely engrossed in their own experimentation. They had few complaints about site support, even though they generally appeared to request or expect much of it. Teachers who reported the impact of the "reflective" researcher tended also to "hang out" together, share the same instructional values and beliefs. They also tended to be among the more "active" teachers at the school, e.g., working on school and district committees, voluntarily and eagerly sharing their successful experiences.

General Comments on the Four Models

The first two categories of support are top-down, and, for the MTS project, a real disaster because: 1) they don't model the student-centered approach at all; 2) they hoard expertise, which is the antithesis of this project. On the other hand, they are not time intensive (i.e., costly) and they let the "supporter" deal from his/her own position of strength , e.g., if the specialist knows about laserdiscs, he or she tends to bring them into the teachers' instructional world.

The third approach (Collaborative) is probably one of the best definitions of "mentoring" as we in the universities imagine and intend it to be practiced. For project purposes, it is the most appropriate in that: 1) it starts with the teachers' instructional/curricular needs and interests (and thus, it models student-centeredness with the teacher as "student"); 2) it distributes knowledge, power, and control to the teachers; 3) it is more likely to provide teachers with generative templates for doing their own planning and implementation of other instructional units than are the other two models; and 4) teachers actually teach themselves something (have insights) about teaching too.

The fourth approach (Reflective) is a surprise; but then ethnographic research often turns up surprises! By answering questions intended to reveal their decision-making principles and instructional perceptions, teachers ineluctably "reflect on" those elements of their work and come to clearer understandings of their own ideas. They better "see" the classroom in action and can begin to "formalize" their own lucky or intuitively effective practices (extract principles or understandings of why/how things worked) and discard or modify those less effective ones. The third and fourth categories are labor-intensive (expensive) and require a certain set of beliefs on the part of the participating teachers.

Communicating Information and Opportunities

Midway through Year One, during a project staff meeting, considerable heated talk revolved around the amount of "telling" project staff should engage in with teachers. Some curriculum and training staff members believed it was important for teachers to have gradual, personal insights about how to be student-centered, apparently largely derived from seeing project staff behave toward them in learner-centered ways. Others expressed the belief that that wouldn't happen in a reasonable time period, and that it was "way too subtle." (The Year One final report describes teachers' frustrations about the lack of explicit direction and feedback on accomplishing project goals.) Borrowing a term from sociolinguistics, one of the researchers suggested that project staff needed to "mark" or make explicit the implicit elements of teacher-centeredness in their own behaviors and in appropriate teacher behaviors. One of the results of that comment was the creation of *The MTS Marker*, an occasional in-house newsletter, edited by a curriculum and training staff member, in which teachers' projects were described. This was distributed to teachers with the idea that it marked occasions of appropriate teacher behavior for other teachers to see. Unfortunately, *The Marker* did not explicitly label elements of teacher projects as student-centered, nor tell why they were. Further, it tended to include projects representing quite a range in interpretations of the notion of "student-centeredness." *The Marker* did serve to inform teachers of things other teachers were doing that were being "accepted" and marked as "acceptable." Teachers could, theoretically, read a variety of teacher projects and deduce the common thread, which frankly, appeared to be the integrated use of technology and letting students make choices, however restricted in importance or range those choices might be. In Year One, eight of the eighteen teachers are written up in *The Marker*. In Year Two *The Marker* tended to include more of the first year high school cadre's projects; only two elementary teachers' projects were written up.

Also in Year Two a new in-house, occasional publication was initiated by the site coordinator. Called *The Recycler*, this publication described newly arrived items, opportunities for teachers and students (such as October Computer Learning Month contests), and upcoming optional "training" sessions (such as Spanish LogoWriter).

It is not clear to how *The Marker* and *The Recycler* were interpreted or used by teachers. Teachers were not heard to discuss either publication, and with very few exceptions, articles in *The Marker* about teacher activities were the result of the staff member interviewing the teacher and writing up the activity as reported by the teacher. With one exception, teachers did not seek publication.

Rewards and Incentives in the Environment

As is often the case in school level programs, a number of environmental factors come to' wield significant influence on teachers and their efforts. Sometimes these factors are deliberately arranged positive reinforcements for specific teacher behaviors; often they are unintentional consequences or opportunities resulting from the interaction between the program and existing school climate. In Year One and especially Year Two, a growing number of teachers recognized and responded to a growing number of rewards and incentives in the MTS environment. These rewards included: conference attendance and speaking opportunities, citation in The MTS Marker, requested formal presentations to colleagues, opportunities to "show off" in bulletin board displays, being selected for closer study by the research team.

There also existed a number of incentives and sanctions that helped shape or pressure teachers to "get with the program." These included principal's inclusion of MTS objectives in teachers' Stull[3] objectives, requests for presentations to colleagues on activities and progress, regular interviews about activities and progress, and student complaints and comparisons about technology access. For five relatively uninvolved teachers, these pressures and sanctions did result in increased technology involvement for their students. However, these teachers did not increase their own technological skills beyond that required to carry out the one technology activity that would get them "off the hook" and "on record" as implementing the MTS project. Another of these teachers is fairly close to retirement and does not behave in ways that might indicate interest in changing her instructional strategies or learning new technology skills. She relies on students for technical expertise and on directive support for appropriate activities. One of the teachers finally decided to transfer to another school in the district to escape the pressure to change. One of the other teachers also chose not to return to Loma Vista for 1989-90.

Administrative Support for Change

The Loma Vista principal (there is no assistant principal) was an important liaison between the MTS project and the school staff. As a member of the MTS management team, the principal was privy to support staff concerns and feedback on support efforts, as well as formative evaluation feedback from the research team. The principal took an pro-active role in encouraging teachers to embrace the project goals. She consistently included MTS goals in any statements about school goals; in Year One and Year Two she posted the MTS commitment to "child-centered" instruction in a "technology-rich environment" as a school goal on one of the main bulletin boards at the entrance to the school. The Loma Vista student handbook also included these school goals along with specific emphases on language arts skills (a district emphasis in Year One and Year Two).

The principal made an effort to know where her staff stood in terms of technology and student-centeredness. Where she perceived foot-dragging, she built MTS objectives into teachers' Stull objectives. Where she perceived active leadership, she provided extra assistance and opportunities, e.g., providing substitutes for teacher release time on fairly short notice. The principal clearly demonstrated her endorsement of the MTS project in a number of ways, including voluntarily making formal presentations at computer conferences and quickly learning and using technologies available to her. Beginning in Year One with the placement of a Macintosh on her desk, the principal pulled back some of the memo-writing and correspondence tasks she had previously passed to her secretaries in favor of generating many of those items herself. It's not clear whether this was an advantage or not. Clearly this freed up secretarial time; but a good deal of secretaries' time was occupied with mastering the Macintoshes and software on their desks. Nevertheless, very early in Year One all school correspondence, internal memos, and announcements were word-processed on the Macintosh by the principal or her staff.

Another indication of her support was the patience she displayed in the slow and tedious shift of administrative tasks from manual to computer processing. She spent time learning to use a school record management system, analyzing and critiquing it, working with the technical assistant to modify the system. Though it presented nothing but problems throughout the year, she stayed with it and appears ready to use it consistently in Year Three. The Loma Vista student handbook was also moved onto computer for desktop publishing. The move tied up one of the two office secretaries for several weeks. She had to learn the desktop publishing software and then was faced with design and layout decisions that further slowed things down. When it got to be October and the student handbook still had not even gotten to the printers' office, the principal joked in an MTS management meeting that technology was creating more problems than it was solving. However, in a later meeting with her whole school staff she declared that it was "important that we give ourselves time to learn." The handbook did eventually get completed, and next year will require only minimal revisions to the file.

---

[3] In California, teachers are periodically evaluated by classroom observations conducted by their principal and focused upon a set of agreed upon objectives.

Relative Effectiveness of Mechanisms for Teacher Change: The Case for School Climate and Principal Leadership

The principal's leadership seemed to have the greatest impact on the more reluctant teachers. Even teachers who were not being "Stulled" that year sometimes found themselves in dialogue with the principal about "what they were doing in their classrooms." For those reluctant teachers, the directive approach of staff support (described earlier) seemed the most palatable. It allowed them to do the project with minimal risk-taking and effort. Unfortunately, such an approach did not move those teachers to make any lasting changes or experience any new learning for themselves. If the MTS project were to withdraw from the site, these teachers would very quickly return to their pre-MTS behaviors. However, continued peer, administrator, student, and project staff pressure may yet sway one or two of these teachers. They do continue to meet minimal requirements for involvement and show up at training sessions. It may be that, for these teachers, the "pay off" for making changes is defined in terms of student outcomes. If they perceive students in the more fully engaged MTS classrooms are learning more or faster, or with greater enjoyment, they may find it more compelling than all the pressure and rewards associated with the project to date.

The majority of the teachers at Loma Vista talk and behave in ways that suggest they are interested and eager to become "student-centered, technology-using" teachers. They generate ideas; they try to master new technologies. They demonstrate patience and reasonable expectations for their own efforts. These tend to be the teachers who are trying to change along two dimensions at once: technology integration and student-centeredness. These teachers are most attracted to and successful with the Collaborative approach to support, and seem to appreciate the rewards of peer and outsider recognition for their efforts. They need help in "marking" and defining what they are doing and what the characteristics of "good" applications are. Techniques that provide those sorts of opportunities are most effective for them.

In Year Two, at least five of the eighteen teachers demonstrated a tendency to teach in ways considered solidly "student-centered" by the curriculum and training staff. For some of these teachers the "student-centeredness" of their instruction was an intuitive response to student differences. Their efforts in becoming student-centered focused on becoming more deliberate and in control their instruction, and so more consistent in it. For these teachers, the technology integration presented the greatest challenge. They quickly figured out how to generate "good ideas" and seemed to gravitate toward the consultative and collaborative models. They also tended to make the most use of the SIG support system in Year One. These teachers talked and behaved in ways that indicated a strong identification with and internalization of the goals and methods of the MTS project. As a result of the recognition and reward systems in the school setting these teachers were readily identified by staff and other teachers as the "leaders" in project implementation. For all teachers in this grouping of expertise, the most cost-effective support was the least restrictive and directive: opportunities to attend technology conferences, to get their hands on technologies and materials, and to find regular occasions for sharing work and ideas with peers.

The numerous formal and informal, deliberate and unintentional mechanisms for supporting teacher change were differentially effective. However, it would be extremely difficult, if not impossible, to sort out the impact of each from the others. For example, it would be impossible to distinguish the effects of the principal's behavior from the support efforts of the MTS curriculum and training staff. Without one would the other have been as effective? In fact, it seems to be the case that the richness and variation in teacher support combined to create an "atmosphere" for change in which each teacher could find support mechanisms that worked best for him or her. The depth or thoroughness of the school climate created by the project staff and principal supported a consistent message to teachers (and students) that the MTS project was not going to go away, that people could not be "exempted" from participation, that administrators and others understood and valued the instructional goals and methods advocated by the project staff, that their colleagues found ways of successfully implementing appropriate instructional activities, and that students and teachers were feeling good about being a part of the MTS project.

Make It Happen—Then Make It Stick

Kathryn Kowalczyk
Teacher Education and Computer Center

Abstract

Collaboration of the entire educational family brings Star Schools to the R.I. School of the Future.

Out of 148 nations in the United Nations, the United States ranks 102nd when it comes to the science and math literacy of its youth. In response to this crisis, several major educational change initiatives are being sponsored by the U.S. Department of Education and the National Science Foundation. The introduction and institutionalization of these programs is no easy task.

Make it Happen focuses on the steps involved in prototyping the new curriculum. Items include teacher training, acquiring necessary materials, public relations, parent involvement, and partnerships with industry.

Make it Stick focuses on the cooperative efforts of local preservice teachers, how they were incorporated into the program, and the predicted impact that this experience will have on their classrooms. Implications for ways of introducing training for preservice teachers will be discussed.

A model for the replication of this process in other geographic areas will be provided.

Applesources: Extending and Integrating the Classroom into the 21st Century

Joyce Morris
Junior High School 143
Bronx, New York

Abstract

The "Leaning Tower of Pizza" is not the name of a new Italian restaurant. It is a structure built of paper and tape standing nearly a meter high and a half meter in circumference. This was only one of several structures built by a class of high risk potential dropouts at J.H.S. 143 in the Bronx. Recipients of an 1989 Equal Time Grant, 150 students at the school have been participating in a variety of hands-on science-computer telecommunications experiments. Design, weather, acid rain, genetics, health polls, and surveys have been some of the subjects collaborated on with other students both nationally and in Great Britain through the STARNET and G.E.T.N. (Global Educational Telecommunications Network). Computers were used for word processing, E-mailing, and databasing. Probe software was used to measure heart beat and reaction times, temperature, light, and humidity. Graphic programs were used to relay design prototypes and graph statistical data. Participation in these telecommunications programs has extended our students' writing skills, data analysis, scientific methodology, and interpersonal skills as well as developed an awareness of different lifestyles and values.

Algorithmic Thinking: A Key to Problem Solving

Anita Marie Stacy
Bishop Brossart High School
Cold Spring, Kentucky

Abstract

"The concept of algorithm should now serve as the key concept for mathematics in schools. The heart of computer science is the construction and testing of algorithms" (Engel: *Elementary Mathematics from an Algorithmic Standpoint*, 1984). In this presentation examples of problems that can be used to develop the student's ability to approach problems from different perspectives will be shared. Programs will be given that show different methods of solving a problem already familiar to the students. It is important for students to realize that methods of solutions to problems may not be unique while answers may be. The student should then be encouraged to explore the possibilities for the most efficient approach.

Several examples are shown that illustrate well how algorithmic thinking can be developed within the computer science environment, including

1) Pascal programs that use iterative procedures to calculate squares, cubes, and higher powers of numbers,

2) The Egyptian method of multiplication to demonstrate the concept of what an algorithm is and how various algorithms can apply to one problem,

3) Finding roots of various problems and helping the student to generalize to an nth root solution,

4) Analyzing iterative vs. recursive methods of solving such problems as GCF, factorials, and the Fibonacci sequence,

5) Using different random number generators given by the teacher and having the student design an experiment to test them, and then, having the students write and test their own algorithm for generating random numbers.

This presentation is an outgrowth of a 6-week training program for teachers held in Santa Barbara, California, for the past two summers. Some of the programs and notes from this workshop on Computing and Algorithmic Math are part of this presentation and emphasize the importance of teaching algorithmic thinking and ideas on how this can be successfully implemented in the high school classroom.

Alternatives to Integrated Instructional Systems

Peter Kelman, Ed.D.
Publisher
Scholastic Software

Introduction

At the core of the recent popularity of Integrated Instructional Systems (IISs) is the conviction among politicians, business leaders, various commissions and educators that our present educational strategies are not working for that considerable segment of the school population regarded as "at-risk". However, while there is little disagreement that our educational system is facing problems of unprecedented proportions, many of us are not convinced that offering so called "at-risk" students a steady diet of IIS no fail, "instant education bytes" will either whet their appetites for learning or develop their minds in ways that will assure them, or our nation, of future success.

This paper will strongly criticize Integrated Instructional Systems as being an inappropriate use of computers with students at risk, as well as being wasteful of limited computer resources in schools. It will further criticize the top-down manner in which these systems are being implemented in many school districts. It will then propose alternative models of computer-facilitated education for all students, including those at risk.

The IIS Phenomenon

A careful analysis of confidential sales figures from the educational computing industry suggests that school spending on computer software and hardware has more than doubled over the past two years. Moreover, these figures show that this increase is due primarily to the purchase by cities and large school districts of comprehensive computer-managed instructional systems, commonly known as Integrated Learning Systems.[*] This analysis also reveals the fact that the funding for these purchases often comes from a variety of federal programs aimed at so-called "children at risk."

Led by the Jostens Learning Corporation (the combination of Prescription Learning and Educational Systems Corporation), more than a dozen IIS companies[**] are estimated to have sold in excess of two hundred million dollars in software alone in 1989. This amount equals the amount spent on non-IIS software from all other sources. Add to this a nearly equal amount spent on hardware to support the IIS software. For 1990, if we also add annual maintenance fees and a rapidly increasing number of these installations, it would appear that IIS-related purchases this year will exceed half a billion dollars.

Although there is considerable variation in curriculum focus, pedagogy, and age-range served among the various IISs, it is clear from private discussions with both buyers and sellers of IISs that their fundamental appeal to school administrators is the provision, in most cases, of: a computer-managed comprehensive basic skills training program for at-risk students that is run largely as a pull-out program in which every child spends 45 minutes to an hour every day; a plethora of accountability-oriented computer generated reports for each child, class, school, etc.; a single vendor for purchasing, technical support and staff training; and, in the case of Jostens, even assistance with funding the purchase through arranged financing and/or help in drawing up politically attractive bond issue proposals.

While representatives of the various IIS companies will undoubtedly claim that one or another of the characteristics just cited does not apply to their product, this overall description of their appeal to district administrators remains accurate. This can be ascertained by analyzing where the majority of IIS installations are (elementary schools in districts that emphasize central decision-making, accountability, and uniform coverage of mandated curricula) and observing how they are used (primarily to teach basic math and language arts skills). (Software Publishers Association, 1990)

Some Lessons from the History of Integrated Instructional Systems

Many of us who have been involved in educational reform efforts over the past 25 years and in educational computing for the past ten years, are painfully aware of the dangers of the kind of quick-fix technological solutions for complex educational problems represented by the current generation of Integrated Instructional Systems and their forerunners, mainframe-based CAI. We are particularly wary of such technological fixes being applied to those children who most need a quality education. Thus, we are troubled by the rate at which many big-city and large-district administrators appear to be rushing to spend enormous amounts of money on IISs, despite these systems' narrow focus on rudimentary basic skills, which in isolation are unlikely to prepare students for real success in life.

---

[*] Integrated Learning System is a somewhat misleading term under which some of these systems are marketed in an attempt to focus the customers's attention on the learning that is supposed to occur when they are used. EPIE, in its study of these systems, argues persuasively that it is more accurate to call them Integrated Instructional Systems, since "they do not possess the level of learner-adaptiveness and other features that education's yet-to-be-developed computer-assisted *learning* systems will one day possess." (p.i.1, Sherry, 1990) For the remainder of this paper, we will use the term Integrated Instructional Systems or IISs.

[**] The largest and/or oldest of these companies are Computer Curriculum Corporation (CCC), Computer Networking Specialists (CNC), Computer Systems Research (CSR), Ideal Learning Systems, Innovative Technologies in Education, New Century Education Corporation, The Roach Organization (formerly Control Data's PLATO system), Wasatch Education Systems, and WICAT Education. (Wilson, 1990)

Further, many of us who are advocates of educational computing worry that the current enthusiasm for IISs will soon turn to disillusionment, to the detriment of all educational computing. This is precisely what occurred in the late 1960s and early 1970s. In 1972, the Educational Testing Service carried out a study to determine why educators at that time had not been particularly enthusiastic about the use of computers in education. (Anastasio et al., 1972) The study concluded that the principal reasons were: their high cost, the unreliability of the technology, and the inadequacy of teacher training. Also mentioned as factors in the study were educators' scepticism about results and negative feelings about computers dehumanizing education.

That study was based on educator reaction to the only use of computers in education at that time, which were IIS-like CAI systems, yet the conclusion drawn by many educators, politicians and the media for years to come was that computers, in general, were not effective or appropriate for use in schools. As a result, throughout the 1970s there was very little administrative support or local funding for computer use in schools, despite the availability of a number of mainframe and later minicomputer-based CAI programs, including CCC, PLATO, and Time Share Corporation's (later acquired by Houghton-Mifflin) Dolphin System. Those systems, the progenitors of today's IISs, were bought in small numbers almost exclusively with federal funding targeted for the "disadvantaged."

Fortunately, the advent of stand-alone microcomputers in 1979 revealed and stimulated interest in a whole range of possible uses of computers in education beyond CAI, including motivational learning games, student and teacher productivity tools such as word processors, spreadsheets, etc., database activities, simulations, telecommunications, creativity and publishing tools, and much more. And, although, as with all use of computers in schools, there are really no reliable studies showing conclusively that any of these computer applications have a long-term ameliorative effect on test scores, there is little doubt in the minds of millions of teachers, students, and parents that such uses of computers enrich their learning and their lives.

Recently, however, with the increasing technical reliability of local area networking (LAN), many schools are linking their stand-alone microcomputers, primarily to achieve certain managerial and administrative goals. In itself, such use of networking can have positive effects on education, in particular more cost-effective use of limited hardware and software resources. Nevertheless, some of us worry that in too many schools networking is being used to turn back the clock, to reduce stand-alone microcomputer use and functionality to that of a terminal on a mainframe computer.

This is precisely the mode in which most IISs use computers. In the typical Jostens IIS installation, for example, short, focused student lessons are slowly downloaded from a central CDROM to a microcomputer workstation in advance of the student's arrival in the computer lab. The student completes the lesson in isolation and the "results" (e.g. time on task, progress through the lesson, items answered correctly/incorrectly, number of tries, etc.) are sent to an instructional management system and stored in a student file on a hard disk.

This is fundamentally the way the early mainframe-based CAI programs worked, except that all processing as well as storage was handled by the mainframe. In fact, some of today's IISs, such as CCC and PLATO are merely microcomputer versions of the original mainframe programs developed in the 1960s and 1970s. Others, such as WICAT and Wasatch were originally developed for minicomputers or proprietary microcomputer hardware, but have recently been adapted for use with today's popular microcomputers. And even the Jostens IIS, although developed from the outset as a network-based system that would run on low-end Apple and MS-DOS machines, owes more philosophically to its mainframe progenitors than to the educational applications of computers that have driven educational computing throughout the 1980s.

Indeed, IISs use computers in exactly the same way that the ETS study found to be unpopular with educators in the early 1970s, and it is our fear that when educators determine that today's IISs are similarly ineffective and inappropriate, there will once again be a backlash against all computer use in education. Already, some administrators are beginning to realize that the on-going high costs of these systems may not be justified by the modest short-term test score gains they appear to engender.

A Brief Critique of Studies of IIS Effectiveness

In making their sales presentations, some IIS companies contend that "the research" has shown their systems to be effective in, among other things, raising test scores. However, according to EPIE:

> EPIE would have liked to have been able to include results of independently-conducted, longitudinal, quantitative studies of IISs. Unfortunately, EPIE was unable to identify a body of such studies. This finding is confirmed by Dr. Henry J. Becker of Johns Hopkins University, an expert in research and evaluation methodology, who has investigated several studies to see if any meet the requirements of

scientific design. To date, he has not identified any that meet this criterion. (p.i.3, Sherry, 1990)

Similarly, this author has read and analyzed virtually every study cited by IISs to bolster their sales argument and has found them all to be unpersuasive.

First, a large number of the studies cited were commissioned by, or done in close cooperation with one or another of the IIS companies, and hence are suspect from the outset. Second, the claims made by representatives of some of the IIS companies regarding the results of such studies are, for the most part, not even supported by the studies themselves; this is especially true where results are too weak to be considered statistically significant or where the experimental design does not permit drawing the conclusion claimed.

Indeed, few of these studies were carried out under an experimental design that could validly determine the effectiveness of the IIS treatment. Thus, even those few studies that detected short-term test score gains cannot prove that the IIS treatment was responsible for these gains. For example, many of the studies were carried out on children in the primary grades, a period when test score gains would be expected to result from ordinary cognitive development. Further, in most of these studies, no attempt was made to control for other treatments, including the rest of the school program, which for many of the students undoubtedly included other "at-risk" interventions.

Moreover, the design of those studies in which gains might be attributable to IIS treatment, fail to isolate variables within the IIS (e.g. computer use, instructional design, time on task, frequency of exposure, teacher involvement, etc.), so that it could just as easily be concluded, for example, that any daily systematic program of instruction could have yielded comparable gains.

Finally, given the relatively short period during which the current generation of IISs has been used, none of the studies can measure long-term gains. Yet such gains are by far the most important, most desireable and most elusive result of any "at-risk" intervention.

In light of the criticisms offered above, it is unfortunate that representatives from many of the IIS companies insist upon citing these various studies to support their sales efforts, particularly for IIS use in at-risk programs. Ultimately, this tactic is likely to backfire, when post-IIS test results in these districts fail to meet expectations.

A Critique of IIS Philosophy of Educational Computing

In addition to doubts about costs and effectiveness, an increasing number of concerned educators, including this author, are expressing their doubts about the educational philosophy of most IISs. To us, IISs represent a step backwards, a retreat from the great progress educational computing has made since the days of mainframe-based CAI. Moreover, they embody a model of education and educational decision-making that has persistently failed to attract the support of teachers and other educators since the late 1960s. As such, these systems are likely to fail once again. Our hope is that this failure will not adversely affect all of educational computing.

It is thus important to distinguish the underlying philosophy of educational computing of most IISs from that of the majority of computer-using educators, who have long since progressed beyond mere CAI. Let us begin this by contrasting certain basic assumptions of more progressive educational computing with those of most IISs.

1. Despite our enthusiasm for computer use in schools, most computer-using educators believe that education is, at base, a human endeavor. It is first and foremost an interaction between people (student-teacher and student-student). Machines can be useful as tools, but not as stand-in teachers.

Although IISs theoretically can be used to facilitate student-teacher interaction, in the vast majority of IISs, that is not the case. In most IIS installations, the students work at a remote lab, their IIS lessons are not connected with the on-going classroom instruction, and the classroom teacher has little or no contact with the student, except, perhaps, to receive a report of progress every few weeks. Indeed, many IIS sales representatives position their system as a constantly on-task, tireless instructor "who" will carry out instructional tasks that teachers can't or won't or don't. This is an offensive rationale for IIS use that most computer-using educators find to be both demeaning to teachers and antithetical to a view of education as fundamentally an interaction between people.

2. Most computer-using educators also believe that teachers and students should control computers, not be controlled by them. Computers can be powerful tools of productivity, creativity, and empowerment. All students and all teachers should be provided with access to and training in the use of such tools.

In contrast, most IISs consist of computer-assisted instruction (CAI) that is little more than an efficient and controlling way to train children to master low-level reading and arithmetic skills. Virtually all thoughtful critiques of the use of computers in education over the past 15 years have pointed to CAI as, at best, a marginally useful and largely trivial use of computing power. To that critique, we might add "irresponsible", even "dangerous".

Numerous educators over the past ten years have suggested that there is a clear danger that computers in education will exacerbate the already widening gulf between the advantaged and the disadvantaged.(Papert, 1980; Coburn et al., 1985) Nowhere is that danger more evident than in the unfortunate attempt to train the disprivileged in low level skills through CAI, while middle-class children are encouraged to use computers in school and at home primarily as tools of creative expression, personal productivity and intellectual empowerment. This contrast is an indictment of our society, and should be a warning to well-meaning educators considering the use of IISs with their disadvantaged students.

3. Most computer-using educators believe that when decisions are made concerning computer use in schools, extreme care must be exercised so that educational value is not sacrificed for bureaucratic convenience. Otherwise the children may suffer.

While it is true that networked environments are often easier to manage than labs filled with hundreds of floppy disks and numerous stand-alone computers that require individual booting of programs, there are many ways other than IISs in which to use such networks. (More on that below). Further, although it may be an administrative dream to deal with one vendor who can supply all your needs: hardware, software, service, training, etc., it may turn out to be an educational nightmare when the curriculum is shallow, the pedagogy inappropriate, the student outcomes disappointing and the financial costs astronomical—with some IISs the cost can be up to $150,000 per 32 student installation.

4. Most computer-using educators have learned from painful experience that unless teachers are involved at all levels of decision making concerning computer use in schools, implementation of the computer program will likely fail. (Coburn et al., 1984) Moreover, in light of the recent move toward "school restructuring", it would appear to be politically unwise for district level administrators to impose major curriculum programs on their schools.

Yet this top-down decision making process is precisely how most IISs are being brought into schools. Despite the fact that such a process flies in the face of "site-based decision making" and "teacher empowerment", in IIS adoption after adoption, teacher involvement is minimal and their decision-making role nil.

Based on even a cursory knowledge of the history of educational reform, it is difficult to imagine curricular programs of the magnitude and cost of IISs being accepted by teachers after being imposed on them from above. This is likely to be especially true of the very teachers who are already disposed toward using computers in schools. These are the technology leaders in the school, yet in most cases the IIS purchase decision-making process ignores their interests and expertise. Moreover, in all likelihood, the pedagogy of the IIS runs counter to their educational computing philosophy. And, perhaps, worst of all in their eyes, after the IIS purchase has been made, there will be very little money left for technology purchases for some time to come. Yet without teacher support, especially from the school technology leaders, it is inconceivable that IISs will succeed in achieving the goals for which they have been purchased.

Ten Dubious Reasons Why IISs are Being Bought: A Summary

If IISs represent as dismal a prospect as indicated so far, why are they being bought in such large numbers by large school districts? Below are briefly critiqued ten major reasons, some of which are laudable in intent, but all of which are flawed in practice:

1. The pressure on large district administrators to do something(!) about their at-risk students.

Critique: All too often IIS adoption is a knee-jerk reaction by an administrator under tremendous pressure. Technology solutions to problems happen to be particularly popular at this time. Unfortunately, ultimate disappointment in technology is often proportionate to initially unrealistic expectations for it.

2. The somewhat illusory appeal of one-stop shopping for all the district's computer-based instructional needs, including hardware, software, technical support and training.

Critique: Most schools with IISs find that, in addition to the high annual charges for IIS maintenance, they still must deal with other vendors for their non-IIS computer needs.

3. The misguided attempt to use local area networks to attain both instructional goals and various administrative goals such as reducing the burden of floppy disk maintenance, dealing with one hardware service contract, and eliminating illegal disk copying.

Critique: Buying an IIS to justify the purchase of a LAN to simplify computer management is a situation in which the tail will end up wagging the dog; there are plenty of legitimate, instructional uses of LANS other than IISs and at far lower costs, as will be detailed later.(Gevirtz and Kelman, 1990)

4. The dubious prospect of a "teacher proof", computer-managed curriculum solution, implemented in computer labs run by low paid teaching assistants.

Critique: In IIS sites in which the classroom teacher is really not involved, the teachers' commitment to the IIS is virtually nil, and the value of the IIS to the students, beyond their fascination with any computer-based activity is questionable. Moreover, the use of teaching assistants, few of whom have any education training, often results in students spending time stuck on a lesson with no appropriate human help available.

5. The misleading claim by many IIS companies that theirs is a comprehensive, diagnostic-prescriptive system that will virtually guarantee significant test score gains.

Critique: Although some IISs do offer diagnosis and prescription, most do not, despite their claims. For example, in the most widely used system, (Jostens) students are tested initially and based on the test they are placed in a given lesson. However, after that point, the student merely progresses through a fixed sequence of lessons. There is no on-going diagnosis and prescription; there is no branching or cycling back, based on problems encountered; there is no sophisticated error analysis; there is very little reteaching; indeed, if a student becomes stuck in a lesson, that fact may not be noticed until weeks later, when and if the report is perused by the classroom teacher. This is all very disturbing, particularly when one considers the price tag and the students served, those most at risk.

6. The appeal of federally funded "pull-out" programs that ease the burden of the classroom teacher, by reducing class-size.

Critique: Pull-out programs are known to be of value primarily when student learning in the program is reinforced in their regular classroom. If teachers are alienated by an IIS, this is unlikely to occur. Moreover, the children who are pulled out for "computer time" are also often the children who are pulled out for other at-risk interventions. This has been known to disrupt the childrens' classroom learning and alienate them from their classmates and teacher.

7. A veritable cornucopia of reports to provide documentation for accountability from the classroom level through the district level.

Critique: In many IIS installations there are too many reports, many of which are difficult to understand or use, and most of which arrive too late to be really useful for the classroom teacher.

8. High powered, often misleading marketing and sales campaigns waged by the more well-heeled IIS companies, such as Jostens.

Critique: These campaigns often result in enormous district-wide purchases made at the top with virtually no site-based input or decision making.

9. The siren call of any educational panacea, particularly a high tech solution, that will give the political appearance of a major commitment to education.

Critique: Historically, the administrator who has made the decision to buy into IISs is long gone by the time the disappointing results come in. When that occurs, the new administration is often stuck with throwing out the now useless software, and in the case of proprietary systems, the hardware as well.

10. Administrative decisions made in the absence of knowledge about viable alternative uses of computers in education.

Critique: Such decisions come about when knowledgeable computer-using educators at the school building level are not consulted, and when decisions are made by administrators who have not been immersed in educational computing over the years.

The remainder of this paper will address itself to this last point by briefly describing some of the more promising uses of computers in education that may be offered as alternatives to IISs. It is hoped that this section, in particular, will be read by district level administrators, who will view it as a starting point for a more careful and skeptical consideration of IIS adoption, one that will involve input and decision-making at all levels of the school district.

Computers as Facilitators of Higher Order Thinking Skills

Most thoughtful educators today recognize that unless children have the opportunity to develop the thinking structures necessary to process new information, all the basic skills training in the world will come to naught. Used creatively and appropriately, computers offer a unique opportunity to provide children with interactive environments in which they can develop these higher-order thinking skills.

Numerous software publishers have for years published dozens of programs aimed at promoting student critical-thinking and problem-solving skills. These programs have been used by thousands of teachers with millions of children with great enthusiasm and success. Yet, almost without exception, the IISs eschew such approaches to facilitating higher-order thinking, in favor of CAI aimed at low-level basic skills. Although some IIS companies claim to address higher-order thinking skills, in few of these cases do the claims stand up to scrutiny of their product.

Today, schools wishing the administrative convenience of networking, without buying into the narrow basic skills-focused IIS vision of education, can do so because most educational software publishers are making many of their stand-alone programs available in forms that will run on networks. This includes hundreds of programs aimed at facilitating student higher-order thinking skills. Even owners of most IISs can use (and indeed are using) such programs, either by installing them on the network alongside the IIS curriculum, or by booting stand-alone versions of these programs in the floppy disk drives at the student work station. It should be noted that in most cases, these IIS owners are doing so without the cooperation of the IIS companies, despite the fact that many of these companies advertise "third-party" compatibility.

One approach to using computers to promote higher-order thinking skills with at-risk students is particularly worth mentioning because it also emphasizes the importance of human interaction in a computer-mediated instructional setting. This approach is the Higher Order Thinking Skills (HOTS) Program developed by Dr. Stanley Pogrow of the University of Arizona. (Pogrow, *Phi Delta Kappan*, 1990; Pogrow, *Educational Leadership*, 1990; and numerous other papers and articles by Pogrow on HOTS) HOTS is not a software program; it is a method in which teachers use off-the-shelf software with children in particular ways that systematically build all-important critical-thinking skills.

HOTS uses computers and special teaching techniques, particularly Socratic dialogues between students and teachers, to develop student thinking and social skills, as well as to increase student self-esteem. The HOTS strategy is proactive, rather than remedial. Instead of reteaching an atomized series of out-of-context "basic skills" over and over again, HOTS provides students with the general conceptual skills that enable them then to learn and retain the complex ideas and information found in most curricula and to do so the first time they are taught—by their classroom teacher.

Implemented as a pull-out program with Chapter 1 and mildly impaired learning disabled students in grades 4-6, HOTS has been validated by the National Diffusion Network. Among other documented successes, HOTS students are currently surpassing national averages for basic skills gains in reading and math, and some HOTS students have actually been rediagnosed as "gifted and talented." This is an especially remarkable result because HOTS does not explicitly teach basic reading and math skills; instead it teaches the underlying thinking skills that enable students to confidently handle new and unfamiliar educational and intellectual challenges they encounter in the classroom, on standardized tests, and in life. If there is any IIS program that has been able to match HOTS in terms of independently validated results, particularly results that last over time, this author is unaware of it.

Computers as Vehicles for Creative Expression

Throughout society and in thousands of schools across the country, computers are used as vehicles for creative expression. Computer-based paint, drawing, and printing programs are used by children to express themselves in the graphic arts. Computer-based music composition and performance programs enable students to explore their musical talents. Word processing, classroom publishing, and other writing-based programs provide students with power tools to express their thoughts and ideas in writing.

Yet, none of the IIS programs provide graphics or music programs, and only a few contain even a rudimentary text editor, let alone a child-appropriate word processing or publishing program. Worse, the use of most IISs to process children through endless sequences of CAI lessons chews up so much of a school's limited and precious computer time that there usually isn't room or time for the children to use creative computer tools in the relaxed and contemplative atmosphere such use requires. Thus, it is not enough that a school is able to buy stand-alone creativity programs to use alongside their IIS program, they must also re-prioritize so that students have the right kind of access and time to use these tools creatively.

Moreover, a school that wishes to supplement an IIS with computer-based creativity tools must configure their computer workstations and on-line printers in such a way that individual students can save and print their products at or near their local workstations. In contrast, many IISs are set up so that all saving is to the file server (since all that is saved is records of student CAI activity) and all printing is to a remote and secure central printer (since all that is printed are reports of student CAI activity).

Computers as Tools of Personal and Professional Productivity

Our society, both in the work place and at home, has been transformed by the emergence in the last ten years of personal and professional productivity tools such as word processors, desk top publishing programs, database managers, and spreadsheets. This fact has been recognized by educators across the country for at least the past five years, as school after school has made these power tools available to their students in a variety of ways: many elementary school students learn the fundamentals of keyboarding and word processing, so they can write short stories and reports on the computer; many middle-school students add to their word-processing skills the experience of using content-area databases to actually research interesting questions in social studies and science; at the secondary school level, many students also get the opportunity to use more quantitative productivity tools such as spreadsheets, statistical packages, and function plotters in their business, social studies and math courses; and desktop printing in schools has become so prevalent at all levels that it is often called "Classroom Publishing."

Yet, none of these tools is available on most IISs. Again, while it is possible with most IISs for a school to buy such programs, many of which are now available in network versions, and to use them side by side with the IIS, this is rarely done. Why? First, because the IIS chews up virtually all available computer time. And second, because after investing sometimes hundreds of thousands of dollars in IISs, there is little money (or professional commitment) remaining for other software approaches. With that kind of investment, what administrator can afford to have the computers used for any other purpose, but the one aimed directly at improving test scores?

This may well be the most serious indictment of IISs, since it means that when computer resources are devoted to IIS training on traditional basic skills, the children, particularly those at risk, are being denied the opportunity to learn the productivity skills that they will need in their lives and in almost any line of work in the twenty-first century. This is not unlike the tragic situation in too many Vocational Education programs in which the neediest students are taught outmoded vocational skills on outmoded equipment, thereby dooming them to the most menial, unskilled labor.

The Role of Computers in Cooperative Learning Environments

Collaborative learning is more than an educational buzzword. It is an educational reflection of the emerging work place of the 1990s. As a result of both technological advances and business practices, more and more offices and factories depend on their employees to work cooperatively on every aspect of a project. And, in the more forward-looking schools across the country, collaborative learning is both a means to and an end of many classroom activities.

Just as computers play a major role in collaborative activities in the work place, so too may they play such a role in collaborative education in the school. Two outstanding examples of this are The Kids Network, developed under a National Science Foundation grant by TERC in Cambridge,MA and Earth Lab, developed under various grants by the Bank Street College of Education in New York City.

The Kids Network, now published by National Geographic, has gotten so much press attention recently (e.g. White, 1989) that it will not be discussed here in detail. Suffice it to say, that in this program, students all across the country (and the world) collect real scientific data and send it via modem to real databases, where real scientists use it to study real phenomena. Talk about empowerment, collaborative learning, and developing real-world skills!

Earth Lab is still at the stage of being implemented at a single pilot site, the Ralph Bunche Intermediate School in West Harlem, NY. (Newman and Reece, in Sherry, 1990). There, both in two computer labs and in various classrooms, students, many of whom are severely at risk, work quietly and enthusiastically at Apple IIe and GS computers connected by either a Corvus or an AppleTalk network. On both network systems, teachers and students can communicate written materials, notes, assignments, etc. through an electronic mail feature built into The Bank Street Writer III. In addition, students work on science investigations in small groups and as a whole-school team, by pooling data they collect in "work spaces" that are specifically set up by project on the network to facilitate such collaborative learning activities. In this way, student learning and instruction emulate the workplace by organizing all activities around projects and problem solving, rather than as a sequence of disconnected skill lessons or individual software programs, selected from a network menu.

Students at Earth Lab can also play a real-time network-based simulation game with other students. Finally, they can access any one of dozens of network-compatible educational programs or boot up stand-alone educational software on disk drives at the workstations. These selections may be student initiated or assigned by teachers, often by "leaving" an assignment in the student's mailbox, not as in many IISs by having a student locked into a lesson by the system.

Earth Lab is a dramatic example of how a network environment can be used to promote specific empowering educational objectives and at the same time be open enough to accommodate individual teacher priorities and individual student interests.

Computers as Multiple Modality Learning Environments

Beyond the rhetoric and marketing hype of educational "multimedia" is the important recognition that all children, especially today's MTV generation, learn through many different modalities. The technologies of interactive videodisc and CDROM, with other multimedia technologies to come, are now presenting educators with ways to provide multiple modality learning environments for their students. Nowhere is this opportunity more welcome than for students at risk.

This year, educators are being treated to a cornucopia of new multimedia products including: ABC Interactive's current events videodisc series with titles on Martin Luther King, the Holy Land, and Election 88; National Geographic's history videodisc product GTV, complete with a "map rap"; Scholastic Software's multimedia educational productivity tool Point of View: The Scholastic History Processor; and a number of multimedia authoring systems like Tutor Tech, HyperStudio, and HyperScreen.

Even some of the IISs are getting in on the act, or at least they would like their prospective customers to think they are. Jostens co-funded the development of the Compton's Encyclopedia on CDROM and includes it on the network alongside their IIS. But, like Jostens' questionably appropriate use of CDROM to download their IIS lessonware, this use of technology is more marketing fluff than of educational value . At this time, only a few of the student stations can access the Comptons Encyclopedia. In fact, when you consider it, the cubicle-like environment of most networked computer labs is singularly inappropriate for most multimedia applications. But worse, there is no time for students to really use the Comptons Encyclopedia, given the priority of moving as many students as possible through their CAI lessons, en route to test score improvement for the largest possible number of students.

Moreover, the mere presence of the Compton's Encyclopedia on the network does not mean it will be used in any meaningful way. Unless it is built into the curriculum, unless teachers provide assignments for which students would need to use it, unless the publisher provides guidance for its use in varied settings, the Compton's Encyclopedia, as delivered by Jostens, will be little more than a marketing ploy.

In contrast, all the other multimedia products referred to earlier come complete with lesson plans, connections to the curriculum, and most importantly, easy to use authoring tools that enable students and teachers to create their own multimedia reports, presentations, and lessons, using superb footage from these videodiscs and any others the school might own or acquire. Now that's empowerment!

Computers as a Means of Empowerment

Let us conclude by reaffirming the desirability of using computers in all schools in the ways they are used in society: to empower their users. This means that computers should be used as personal and educational productivity tools by teachers and students. A corollary to this goal is the view expressed in writing and in speeches by countless educational computing leaders, that it is a waste of computing power, financial resources, and human potential to use computers for endless, lock-step atomistic CAI lessons.

Instead, computers should be used by teachers to write lesson plans, reports, handouts, notes to students and teachers; to comment on student written work and other forms of expression; to keep track of student progress in their classroom activities; to facilitate classroom presentations and demonstrations; to keep up with professional developments and share ideas with colleagues, etc.

Thus, teachers need a computer workstation, loaded with personal and professional productivity software, at their desks, linked via network to student workstations in the classroom or lab. And they need a similar arrangement at home, linked to the school by modem. Is this vision too futuristic? Not if we regard it as a guiding vision that we can work toward gradually. Many of us prefer such a vision, even if it cannot be fully implemented today, to the spectre of IIS administrator-controlled, centralized, record-keeping systems with little or no direct access by teachers. Apparently, so do the major teachers' organizations, to judge from a number of Albert Shanker's recent speeches and the NEA's request for information from the hardware companies for an affordable teacher workstation, similar to that described above.

Like their teachers, students should have the opportunity to use computers to explore ideas, to analyze data, to write papers and stories, to receive and carry out written assignments, to research databases, to design, compose music, publish, plan, work with a team on projects, etc.

Thus, students should have almost constant access to a computer workstation in the classroom and the lab, linked via network, to a fileserver containing a vast array of productivity software, creativity software, wholesome recreational software, educational software, shared hardware resources, shared databases, telecommunications capabilities and much more.

This vision is not futuristic in the least. All of this is possible and is in place today in schools that prefer this vision of education (and networking) to the narrow vision of computer labs filled with students working in 20 minute chunks of time on "byte-sized" lessonware, controlled by an impersonal lesson sequence, developed by a distant software publisher.

References

Anastasio, Ernest and Morgan, Judith; *Study of Factors that have Inhibited a More Widespread Use of Computers in the Instructional Process*; EduCom, Interuniversity Communications Council; 1972.

Coburn et al.; *Practical Guide to Computers in Education*, Second Edition; Addison Wesley; 1985.

Gevirtz, Gila and Kelman, Peter; *The Scholastic Guide to Educational Computer Networks*; Scholastic; 1990.

Newman, Dennis, and Reece, Paul; "Using a Local Area Network for Sharing Data," in Sherry; 1990.

Papert, Seymour; *Mindstorms: Children, Computers, and Powerful Ideas*; Basic Books; 1980.

Pogrow, Stanley; "A Socratic Approach to Using Computers with At-Risk Students," in *Educational Leadership*; February 1990.

Pogrow, Stanley; "Challenging At-Risk Students: Findings from the HOTS Program," in *Phi Delta Kappan*; January 1990.

Sherry, Mark; *EPIE Institute's Report on Computer-Based Integrated Instructional Systems*; 1990.

Software Publishers Association; *Education Vendors Survey On Present and Projected Use of Networkable Software and Integrated Learning Systems in Schools*; 1990.

White, Mary Alice, "Educators Must Ask Themselves Some Important Questions," in *Electronic Learning*; September 1989.

Wilson, Judy; "Integrated Learning Systems: A Primer," in *Classroom Computer Learning*; February 1990.

Working Together:
Increasing the Effectiveness of Collaborative Computer Based Learning Groups

Judi Repman
College of Education
Box 4560
Texas Tech University
Lubbock, TX 79409-1071
(806) 742-2394

Abstract

Students who use computers in school settings frequently work in groups. Research clearly indicates a positive relationship between the kinds of verbal interactions that take place in these collaborative learning groups and student achievement. Giving elaborated explanations to other group members is the verbal behavior that is most often associated with increased achievement, yet it is a type of behavior that many students do not engage in.

This paper reports on the results of a study conducted with 190 seventh grade social studies students. The purpose of the study was to investigate the effectiveness of varying levels of structure in collaborative learning groups. Intact classes were randomly assigned to receive one of three treatments: unstructured collaboration, structured collaboration, and structured collaboration with training. Students who received structured collaboration with training gave the most explanations during the collaborative learning sessions. Providing structure (with or without training) resulted in increased content area achievement.

Introduction

Collaborative learning is one instructional technique that has been frequently suggested as an effective and efficient alternative to traditional patterns of classroom organization. When faced with a limited amount of computer resources, teachers often allow students to work in groups on computer based learning assignments. Johnson and Johnson (1985, p. 13) suggest that collaborative computer based learning "promotes more and better work, more successful problem solving, and higher performance on factual recognition, application and problem-solving tasks".

At the same time, the pattern of collaboration that is frequently observed involves one student "expert" who takes an active role in entering information or experimenting with different tactics to solve the problem, while the remainder of the group acts as passive observers (Diem, 1986). Structuring collaborative learning groups by providing students with specific roles and responsibilities is a simple and inexpensive technique that can be successfully used to minimize passive behavior and encourage verbal interaction. Training in effective group processing could be used to provide students with models of giving explanations in response to questions or errors.

Theoretical Framework

Verbal Interaction and Achievement

General findings concerning the effectiveness of collaborative learning have been widely published. In a recent review of 60 studies, Slavin (1989) found that 72% of the studies reported positive outcomes for cooperative learning, with only 8% showing outcomes favoring control groups. Webb (1982a, 1982b, 1985, 1989) has extended this research, focusing on specific student verbal behaviors that occur in collaborative settings.

Among the types of verbal interactions identified through this research, group helping interactions (that is, giving or receiving help) have generally been found to be associated with increased achievement (Webb, 1982c). When a student asks for help they may receive an explanation, terminal help (the correct answer without an explanation), or no help at all. Receiving no help or a terminal explanation has been found to be negatively correlated with achievement, while receiving elaborated explanations is positively related to achievement (Webb, 1985). Additional research indicates that when students are engaged in higher-level thinking or problem solving activities with the computer their success is related to the number of task-related questions asked and the amount of time spent on strategy development and elaboration (King, 1989).

Providing Structure and Training

Providing scripts or protocols to guide students engaged in collaborative learning has been found to be a successful technique (Dansereau, O'Donnell, & Lambiotte, 1988; Swallow, Scardamalia, & Olivier, 1988; Yager, 1985). Specific roles that have been suggested for use in collaborative, computer based learning include keyboarder, recorder, checker, and encourager (Johnson & Johnson, 1985).

Training students to give help and ask for help (especially elaborated help, or explanations) has been identified as one way to significantly improve the effectiveness of collaborative learning (Bossert, 1988-89; Newman & Thompson, 1987; Webb, 1988). This may be particularly important for the kinds of critical thinking and problem solving tasks students are exposed to as part of the social studies curriculum (National Council for the Social Studies Task Force on Scope and Sequence, 1984).

Research Questions

This study was designed to answer two research questions:

1. Does the additional of structure (with or without training) to collaborative computer based learning groups result in improved content area achievement? and

2. Is training an effective way to increase the number of explanations given (in response to errors or questions) in collaborative computer based learning groups?

Methodology

The Sample

Nine intact classes (N = 190) of seventh grade social studies students at two middle schools took part in this project. The sites were selected because they contained large populations of students at-risk of school failure. 56% or the participants were identified as at-risk of school failure and 81% were members of a minority group. None of the teachers or students involved in this research had used computers as part of their social studies instruction.

Procedures

The curriculum. As part of this project, the researcher worked with the participating teachers to identify instructional objectives and thinking skills to be taught during weekly computer lab sessions. The computer based learning activities were designed to enhance, not replace, the instructional program already in place. Software utilized during the course of the research included *Bank Street Writer* (Scholastic), *pfs: File* (Scholastic), *Where in the U.S.A. is Carmen San Diego?* (Broderbund), and *Ten Clues* (Sunburst).

The treatments. Intact classes were randomly assigned to receive one of three treatments: unstructured collaboration, structured collaboration, and structured collaboration with training. During the nine-week experimental period each class spent one class period per week in the computer lab, completing the researcher-prepared activities described above.

In each treatment, students were assigned by their teachers to groups of three. In Treatment 1, unstructured collaboration, students were told to work together to complete the computer based task but were given no instruction on the collaborative learning process itself. Students were also instructed to take turns keyboarding during their computer session.

Students in Treatment 2, structured collaboration, received the same materials and used the same software as students in Treatment 1. They were also provided with a protocol sheet to guide their collaboration in the computer based learning activities. An example of a protocol sheet is included in Figure 1. While the roles remained the same throughout the experimental period, the responsibilities were tailored to fit specific assignments. For the example shown, the students were creating a database of information on the fifty states. Students were told to select a different role each week.

For Treatment 3, structured collaboration with training, students received instructional materials identical to the other treatments along with the same collaboration protocols used in Treatment 2. They also took part in three 50-minute sessions of training in collaborative learning. The training, conducted by the researcher using tasks similar to those used during the computer based learning sessions, focused on the use of elaborated explanations in response to questions and errors.

Instruments. To assess differences in content area achievement students were pre- and posttested using the Comprehensive Test of Basic Skills, Social Studies Subtest, Level H Form U (Monterey, CA: CTB/McGraw-Hill, 1983). This test contains 40 multiple choice questions pertaining to geography, economics, history, political science, sociology, and interdisciplinary studies.

Observable changes in student behavior resulting from the treatments were investigated through classroom observation. Three groups of students from each class were selected randomly for four five-minute observation periods during the computer based learning sessions. Student verbal interactions were recorded and coded into three categories: giving explanations, giving input suggestions, and total errors. Explanations include statements such as "You do this because..." or "I think the reason is...". Input suggestions are limited to comments such as "Hit the space bar" or "You forgot to put a period there".

Results

Content area achievement

Pre- and posttest data from the CTBS Social Studies Subtest was analyzed using univariate analysis of covariance (with the pretest scores serving as the covariate). Although no significant difference was found from the overall analysis ($F(2,124) = 2.90$, $p < .06$) a planned orthogonal contrast found a significant difference ($F(1,124) = 5.18$, $p < .02$) in favor of the structured and training conditions when contrasted with the unstructured condition. The least squares means (which have been adjusted by the use of the covariate) for the CTBS Social Studies Subtest are shown for each treatment group in Table 1.

Verbal interactions

Observational data was collected and coded for a randomly selected sample of the groups participating in the study. The scores for each group on explanations

THINK--LISTEN--DISCUSS

WORKING TOGETHER

The Keyboarder

1. Enter the information needed to complete each field of the record for your group's state. Read the information OUT LOUD as you enter it.
2. Make sure you are putting the CORRECT information in the CORRECT field. If you do not understand something ASK your group to EXPLAIN it to you.

THINK--LISTEN--DISCUSS

WORKING TOGETHER

The Questioner

1. Find the information you need to complete your record on the state data sheet given to your group.
2. Make sure that you EXPLAIN to the rest of your group WHAT information you need to complete your record and WHY that information goes in a certain field.

THINK--LISTEN--DISCUSS

WORKING TOGETHER

The Checker

1. Make sure that your group is working CAREFULLY.
2. Is the information you are entering CORRECT? Is the keyboarder putting the information into the CORRECT field? If you disagree with your group's decisions, EXPLAIN your reasoning to the rest of your group.

Figure 1. Sample Collaboration Protocol.

given, input suggestions made, and total errors were summed across four observations to provide the three dependent measures used in the analyses of variance. Table 2 shows the means by treatment group for each dependent variable. Significant differences were found for the number of explanations given ($F(2,19) = 4.96$, $p < .02$). No differences were found between the treatment groups for the number of input suggestions made or number of total errors.

Discussion

This study demonstrates that providing students in collaborative computer based learning groups with a combination of structure and training results in increased content area achievement, a finding similar to other research (Dansereau, O'Donnell, & Lambiotte, 1988; Swallow, Scardamalia, & Olivier, 1988; Yager, 1985). As demonstrated in previous research (Dansereau, 1988; Yager, 1985), the effects of structured collaboration transferred from small-group

Treatment	n	LS Mean	SE
Unstructured	39	19.81	.93
Structured	45	22.80	.84
Training	44	21.94	.86

Note. Maximum score = 40.
SE = standard error LS mean.

Table 1. Least Squares Means for CTBS Social Studies Test

learning to individual performance on an achievement test.

Training in giving explanations as part of the collaborative learning process was also shown to be an effective method of increasing the number of explanations given in relatively homogeneous low-ability learning groups. No significant differences were found between the three treatment groups for the other verbal interactions of interest in this study (making input suggestions—a low-level verbal interaction) or in the number of errors observed, justifying the conclusion that the number of explanations given differed as a result of the training that the students received.

It should be noted, however, that even though students in the training classes gave explanations more frequently, the rate was still fairly low (an average of slightly more than one explanation during each observed session). Further, several of the groups receiving training gave *no* explanations during one or more of the sessions that were observed. One possible explanation for the lack of a direct effect for training on social studies achievement is that despite an increased rate of giving explanations, students who received training still did not engage in this behavior frequently enough to produce measurable results. Shorter training sessions, given more regularly (weekly or even more often), might have resulted in more consistent and frequent use of explanations within the learning groups. At the same time, this study clearly demonstrates that low-ability students who receive no training in providing higher level elaborations almost *never* engage in this behavior.

Results from this research indicate that collaborative, computer based learning within social studies classes appears to be a promising combination of instructional methodologies for at-risk and regular middle school students. Enhancing the collaborative learning process through the use of structure and training does result in increased use of elaborated explanations and promotes student engagement in critical thinking and problem solving activities within the social studies content area.

Although most computer assisted instruction continues to be devoted to drill-and-practice activities, the computer is also a powerful tool for the delivery of critical thinking and problem solving activities within the social studies (Budin, Taylor, & Kendall, 1987; Lengel, 1987). For tasks where higher order thinking is necessary for success, collaborative learning provides students with the opportunity to be aware of not only their own thinking, but also the thinking process of their peers (Forman, 1981; Resnick, 1987; VanSickle, 1982). Collaborative learning techniques are not expensive to implement, and it is the belief of this researcher that they can be used to enhance traditional classroom instruction as well as instruction in alternative environments such as computer labs.

References

Bossert, S.T. (1988-89). Cooperative activities in the classroom. *Review of Research in Education, 15*, 225-250.

Budin, H., Taylor, R., & Kendall, D. (1987). Computers and social studies: Trends and directions. *The Social Studies, 78*, 7-12.

Dansereau, D.F. Cooperative learning strategies. In C.E. Weinstein, E.T. Goetz, & P.A. Alexander, (Eds.), *Learning and study strategies: Issues in assessment, instruction, and evaluation* (pp. 103-120). New York: Academic Press.

Treatment	n	Dependent Measure		
		Explanations	Input Suggestions	Errors
Unstructured	7			
M		2.86	14.29	2.86
SD		1.95	3.25	2.43
Structured	7			
M		3.57	13.57	2.86
SD		1.99	4.61	1.60
Training	8			
M		5.75	13.50	2.00
SD		1.67	2.67	1.70

Table 2. Means for Verbal Interactions and Errors by Treatment

Dansereau, D.F., O'Donnell, A.M., & Lambiotte, J.G. (1988, April). *Concept maps and scripted peer cooperation: Interactive tools for improving science and technical education.* Paper presented at the annual meeting of the American Educational Research Association, New Orleans, LA.

Diem, R.A. (1986). Computers in a school environment: Preliminary report of the social consequences. *Theory and Research in Social Education, 14*, 163-170.

Forman, E.A. (1981. The role of collaboration in problem-solving in children. (Doctoral dissertation, Harvard University, 1981). *Dissertation Abstracts Interantional, 42*, 2563B.

Johnson, D.W., & Johnson, R.T. (1985). Cooperative learning: One key to computer assisted learning. *The Computing Teacher, 13* (2), 11-13.

King, A. (1989). Verbal interaction and problem-solving within computer-assisted cooperative learning groups. *Journal of Educational Computing Research, 5* (1), 1-15.

Lengel, J.G. (1987). Thinking skills, social studies, and computers. *The Social Studies, 78*, 13-16.

National Council for the Social Studies Task Force on Scope and Sequence. (1984). In search of a scope and sequence for social studies. *Social Education, 48*, 249-262.

Newman, F.M., & Thompson, J.A. (1987). *Effects of cooperative learning on achievement in secondary schools: A summary of research.* Madison, WI: Wisconsin Center for Education Research. (ERIC Document Reproduction Service No. ED 288 853)

Resnick, L.B. (1987). *Education and learning to think.* Washington, D.C.: National Academy Press.

Slavin, R.E. (1989). Cooperative learning and student achievement. In R.E. Slavin (Ed.), *School and classroom organization* (pp. 129-156). Hillsdale, NJ: Lawrence Erlbaum.

Swallow, J., Scardamalia, M., & Olivier, W.P. (1988, April). *Facilitating thinking skills through peer interaction with software support.* Paper presented at the annual meeting of the American Educational Research Association, New Orleans, LA.

VanSickle, R.L. (1982). A social perspective on student learning. *Theory and Research in Social Education, 10* (2), 21-31.

Webb, N.M. (1982a). Group composistion, group interaction and achievement in cooperative small groups. *Journal of Educational Psychology, 74*, 475-484.

Webb, N.M. (1982b). Peer interaction and learning in cooperative small groups. *Journal of Educational Psychology, 74*, 642-655.

Webb, N.M. (1982c). Student interaction and learning in small groups. *Review of Educational Research, 52*, 421-445.

Webb, N.M. (1985). Verbal interaction and learning in peer-directed groups. *Theory into Practice, 24* (1), 32-39.

Webb, N.M. (1988). The social context of learning computer programming. In R.E. Mayer (Ed.), *Teaching and learning computer programming: Multiple research perspectives* (pp. 179-206). Hillsdale, NJ: Lawrence Erlbaum.

Webb, N.M. (1989). Peer interaction and learning in small groups. *International Journal of Educational Research, 13*, 21-39.

Yager, S.O. (1985). The effects of structured oral discussion during a set of cooperative learning lessons on student achievement and attitude (Doctoral dissertation, University of Iowa, 1985). *Dissertation Abstracts International, 46*, 1588.

Embedding Learning About Women's Health Care in Computer-based Simulations

Joyce E. White, R.N., Dr.P.H.
Family Nurse Practitioner and Assistant Professor
Primary Health Care Nursing Graduate Program
University of Pittsburgh School of Nursing

Abstract

Problem-based learning involves students in acquiring and manipulating data which are presented within the context of "real life" situations. Problem-based learning is valued because it increases skills in problem solving; manipulating, and not just acquiring data.

At the University of Pittsburgh we have developed an approach to integrating more problem-based learning into the Primary Health Care Nursing (PHCN) curriculum through the use of computer-based simulations. This approach offers the opportunity for students to integrate theoretical and foundational knowledge into patient care simulations.

Two of the simulations are described here. The first simulation is that of a middle-aged woman who has come for a routine checkup. Students are expected to take her health history and develop a health risk profile. This activity is required as part of the health promotion course which students take as a first course in the Primary Health Care Nursing clinical sequence. The second simulation is designed to teach students the assessment of a patient who wishes to initiate oral contraceptive use and is part of the first management course which students take in the PHCN sequence.

The simulations are developed using Precept and they run interactively with videotape using equipment manufactured by BCD, Inc.

Introduction

Problem-based learning has been described as requiring the learner to focus on problems which require genuine intellectual effort, to think rationally about them, to pursue a systematic search for information and to synthesize a problem solution. The advantages of problem-based learning include enhanced retention and transfer of information, heightened motivation and systematic thinking (Barrows, 1984; Barrows and Tamblyn, 1979). McMaster University of Hamilton, Ontario was an early proponent of a curriculum built on problem-based learning and many of the strategies currently in use to increase problem-based learning were developed there (Neufeld and Barrows, 1974).

The faculty who teach in the Primary Health Care Nursing graduate program at the University of Pittsburgh School of Nursing are increasing problem-based learning within a traditional curriculum which prepares family and adult nurse practitioners using computer-based interactive video patient simulations.

Background

Problem-based learning in the health sciences requires the development of a number of clinical presentations which involve the student in learning about patient care within the context of providing patient care. Although the most "real" of these clinical presentations is no doubt the actual patients whom students encounter in their clinical placements, simulations which mimic these real patients are useful as students organize the many domains of knowledge which are involved in learning to care for patients. For example, presentation of a patient with chest pain facilitates learning about cardiac physiology, signs and symptoms of myocardial ischemia, appropriate diagnostic testing, pharmacologic therapies and meeting comfort and educational needs of patient and family.

Before powerful, relatively cheap microcomputers became so available to students, presentation of clinical cases involved patient management problems (PMP's), Portable Patient Problem Packs (P-4 Deck), and simulated patients. The PMP's are paper and pencil simulations. They present brief descriptions of the patient's presenting complaint(s) and then provide additional information as it is requested by the student. This usually involves the students' using a special device to make the data visible, a process called latent imaging (Holzemer, Schleutermann, Farrand, Miller, 1981; Goran, Williamson, Gonella, 1973).

The P-4 Deck consists of a number of cards. The student begins with the situation card, then goes on to select cards on which are typed the questions and requests for physical or laboratory data which most closely approximate the information the student wishes to acquire. The patient's responses, physical exam findings and lab results are typed on the back of the cards (Barrows and Tamblyn, 1977).

Simulated patients are actors who portray the case histories of actual patients. In some instances, they also replicate the physical findings of the actual patient. For example, simulated patients used for problem-based learning in the area of neurological diseases have been taught to mimic a flaccid paralysis below the spinal cord level at which the actual patient had experiences a cord transection (Barrows, 1971).

SIMULATIONS:

1. makes the learner a participant in a realistic learning experience.
2. provide access for certain problems not readily available to the learner.
3. allow reproducibility of the same problem either for the same learner until mastery, or for different students, facilitating the evaluation process.
4. allow compression of time; the status of the patient can be changed in a few minutes, although it may take years in real life.
5. eliminate risk to the patients.
6. provide feedback to the learner about his performance, in the learning and evaluation situations.

Figure 1. Educational Advantages of Patient Simulations

All of these approaches to simulation provide a number of educational advantages (see Figure 1). They permit more active learning in an environment in which the student can receive immediate feedback and in which patient safety is not compromised (Norman, Painvin, 1987).

The use of computer-based simulations as ways to present the problems in which learning is imbedded has developed as computers have become more available and easier to use and as faculty have had increased access to software and authoring systems. Using computers to present patient simulations was preceded by a number of years of increased use of computers in nursing education (Belfry and Winne, 1988). They were largely used, at first, in ways that teaching machines and programmed texts had been used; that is, to present information and to test its acquisition (Birgham and Kamp, 1974), in what is sometimes referred to as "drill and practice". These early efforts at computer assisted instruction have also sometimes been labeled "electronic page turners" because they simply presented text on a computer screen. Learners did nothing more than page through the text as they would if they had been using a textbook (Abdulla, Henke, Watkins and Lindsay, 1985).

As microcomputers proliferated and as evidence grew that computer assisted instruction (CAI) was an effective teaching strategy (Larson, 1981; Kulik, 1983), its appropriateness as a way to present patient simulations became apparent. The use of such simulations has expanded rapidly in nursing. Hebda found that approximately 65% of the CAI packages which were being used in baccalaureate nursing programs were simulations (Hebda, 1988).

Early simulations in both nursing and medicine were patterned after early Patient Management Problems. These PMP's were not found to be highly reflective of actual patient care, perhaps because their format involves "cueing" of the student in a way that the actual clinical situation does not. Newer simulations of the type developed by the Primary Health Care Nursing graduate faculty avoid this cueing and have been used successfully to teach such sophisticated skills as health history taking (Nardone, Schriner, Guyer-Kelly and Kositch, 1987). Interactive video can be expected to replicate the clinical setting to a much higher degree than does the computer alone (Kramer, T.A., Polan, 1988). Students using our simulations, for example, may watch the patient's face as he/she responds, listen to the auscultation of the heart and examine sinus x-rays.

Simulations in the PHCN Program at the University of Pittsburgh

When faculty in the Primary Health Care Nursing Graduate Program decided to use computer-based simulations, they had already had several years experience with simulated patients (McDowell, Nardini, Negley and White, 1984). The decision to move to computer-based simulations was made on the base of cost of recruiting and training patient simulators each term.

Before embarking on what was to become a three-year search for an appropriate authoring system, the faculty had defined characteristics of the simulations they wished to develop. These characteristics are believed to increase the degree to which the computer-based simulations replicate the actual clinical setting. Figure 2 lists these charasteristics.

The faculty selected Precept, an authoring system designed specifically for the creation of clinical

SIMULATIONS:

1. should not involve cueing through the use of menus or other devices.
2. should permit querying with standard English.
3. should not impose an artificial linearity which forces the student to encounter the patient in a way that she/he would not in the clinical area. By this, the faculty meant that there should be no sequence imposed on data gathering; as in actual clinical practice the student should be able to ask a question, do a piece of the physical exam and ask another question.
4. should, with minor modifications, be usable for both teaching and evaluation.
5. should support interactive video.

Figure 2. PHCN Faculty-mandated Computer Simulation Characteristics

UNIVERSITY OF PITTSBURGH SCHOOL OF NURSING

PROBLEM LIST and HEALTH RISK PROFILE

#	ACTIVE PROBLEMS	INACTIVE PROBLEMS
	HEALTH RISKS	AT RISK FOR

Figure 3. Problem List and Health Risk Profile

simulations, to aid in lesson development. Authoring systems permit the development of sophisticated computer lessons by faculty who are experts in nursing, but who lack sophisticated programming skills by creating a "shell" into which faculty place content. Widely described in medical education literature (Abdulla, Henke, Watkins, Lindsay, 1985; Abdulla, Henke and Watkins, 1984; Abdulla, Watkins, Henke, 1983), Precept guides the author of the lesson through the tasks involved in constructing a lesson; entry and editing of lesson material in the form of text files, converting text files to a format which can be used to create lesson discs (compiling) and transferring the compiled program to lesson discs which are then ready for student use designed to run interactively with videodisc. Developer Henke modified our version to run interactively with videotape when we found videodisk development to be too expensive. Information about Precept is available from CME Systems, Inc., of Augusta, Georgia. Information about the equipment used to achieve videotape interaction can be obtained from B.C.D. Associates, Oklahoma City, Oklahoma.

All simulations are based on real patients, usually drawn from the caseloads of PHCN faculty, all of whom maintain clinical practices as part of their faculty responsibilities. Faculty rejected the use of "textbook cases" because they occur so rarely in actual clinical practice.

Two lessons created by PHCN faculty will be described here. The first of these is designed to operationalize the concepts from health promotion in which the likelihood of a specific patient's developing a particular disease or condition is assessed on the basis of the patient's having risk factors for that particular disease or condition. These risk factors may be biologic in nature, that is, related to the patient's genetic inheritence (for example, a woman whose maternal aunts developed breast cancer before menopause is at increased risk of developing breast cancer herself) and personal history of disease (an individual with elevated blood pressure is at increased risk for a heart attack or a stroke.) Risk factors may be related to lifestyle, such as the patient's use of tobacco and alcohol, or related to the environment, which includes occupational exposure to substances such as lead, cotton fibers and coal dust. In addition, the patient's utilization of the health care system places her/him at varying levels of risk; the woman who never has recommended pap tests is less likely to have cervical cancer detected early.

In the first simulation a middle-aged woman is introduced to the student as seeking a "check-up". The student proceeds to gather a health history, typing in questions in English using normal syntax. The student may acquire physical examination data by asking for parts of the exam, such as "heart exam", "abdominal exam". Results of the exam may be displayed on the computer screen or presented to the student via videotape. As the student completes each section of the

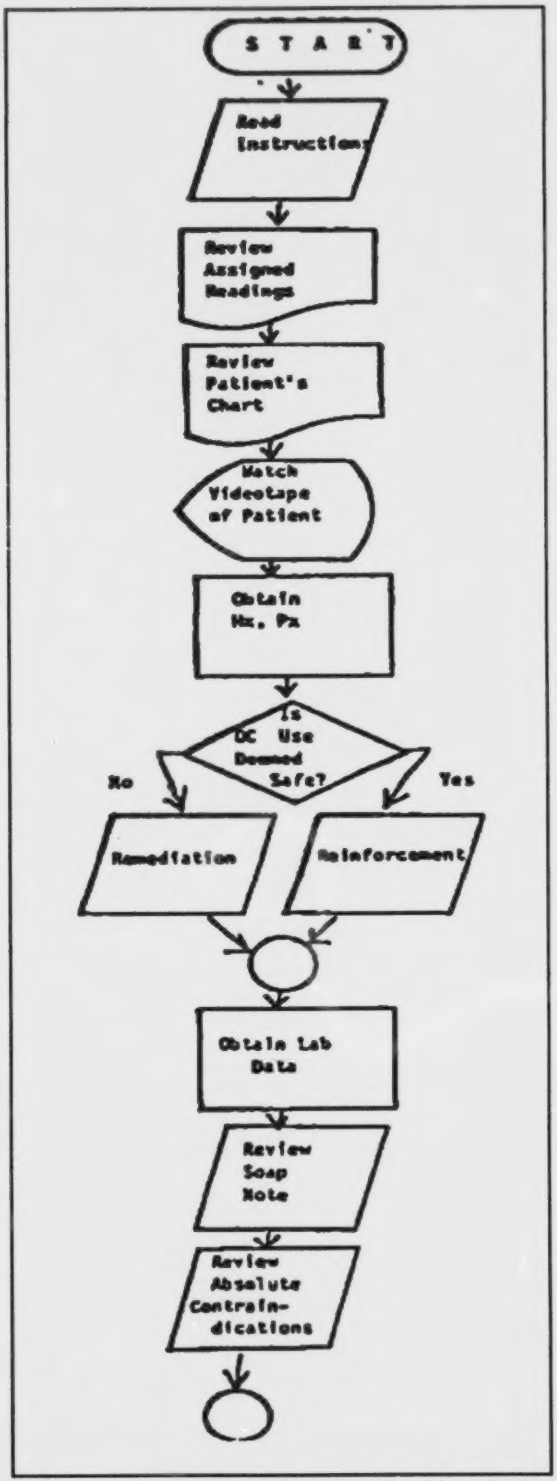


Figure 4. Flowchart of Program Teaching Patient Assessment for Oral Contraceptive Use

typical health history, he/she may be given feedback about its adequacy. Students may also be given feedback on their ability to create a problem list and health risk profile for the patient by the instructor. In this case, the student develops a health risk profile and problem list just as she would for an actual patient and hands it in to the instructor using the form shown in Figure 3.

The second lesson described here is designed to teach students the process of evaluating a patient for oral contraceptive use. The patient is a young woman who is seen, on videotape, asking as the student begins the program, "Is it okay for me to use birth control pills?" In answering this young woman's question, typical of questions asked every day of nurse practitioners in practice, the student uses reference books, refers to the patient's "chart" which contains information about past health history and family history, and interviews and examines the patient via computer. The lesson provides feedback and remediation where needed, shows the student what information would be included in the progress note completed for this patient by an experienced clinician and finishes with a review of absolute contraindications to oral contraceptive use. Figure 4 is a flow sheet for this program.

These lessons and others designed by the faculty use an interactive work station which includes an IBM personal computer with a hard drive and a Sony VCR using VHS format. Although videodisc offers the advantage of reduced time as the machine searches for the specific video segment needed to answer the student's question, faculty have been satisfied with the less expensive interactive videotape which has the added advantage that it allows for ease of editing should the faculty with to change or add segments. It also increases the potential for "in-house" production. We have dealt with the problem of longer search time by placing the segments in the sequence in which students are expected to access them and by providing some text for students to read while the VCR is searching.

In conclusion, PHCN faculty has found the time and effort involved in developing interactive computer simulations to be compensated by its ability to increase the opportunities for problem-based learning, a type of learning which we believe to be most effective in teaching the skills needed by graduates of our Primary Health Care Nursing Program.

References

Abdulla, A.M., Henke, J.S., Watkins, L.O., Lindsay, M.D. (1985). Computer-aided learning using a laser videodisc player. *Physicians and Computers Journal.* 2(10), 38-44.

Abdulla, A.M., Henke, J.S., Watkins, L.O. (1984). Microcomputer-assisted problem-based learning. *Resident and Staff Physician.* 30(9), 88-102.

Abdulla, A.M., Watkins, L.O., Henke, J.S. (1983). Computer assisted clinical simulations in continuing medical education. *Physicians and Computers.* 1(1), 32-35.

Barrows, H.S. (1985). *How to design a problem-based curriculum for the preclinical years.* New York: Springer Publishing Co.

Barrows, H.S. and Tamblyn, R. (1979). *Problem-based learning in health sciences education.* Atlanta: U.S. Department of Health, Education and Welfare.

Barrows, H.S. and Tamblyn, R. (1977). *Instructions on the use of the P-4 System.* Hamilton, Ontario: McMaster University faculty of health sciences (unpublished).

Barrows, H.S. (1971). *Simulations; the development and use of a new technique in medical education.* Springfield, Ill.: Thomas Publishing Company.

Belfry, L.N. and Winne, P.H. (1988). Review of the effectiveness of computer assisted instruction in nursing education. *Computers in Nursing.* 6, 77-85.

Brigham, C.R., and Kamp, M. (1974). The current status of computer assisted instruction in the health sciences. *Journal of Medical Education.* 49, 270-279.

Goran, M.J., Williamson, M.D. Gonnella, J.S., (1973). The validity of patient management problems. *Journal of Medical Education.* 48, 171-177.

Hebda, T. (1988). A profile of the use of computer assisted instruction within baccalaureate nursing education. *Computers in Nursing.* 6, 22-29.

Holzemer, W.L., Schleutermann, J.A., Farrand, L.L., Meiler, A.G. (1981). *Simulations as a measure of nurse practitioners problem solving skills.* Nursing Research, 30(3), 139-144.

Kramer, T.A., Polan, H.J. (1988). Use and advantages of interactive video in medical training. *The Journal of Medical Education.* 63, 643-644.

Kulik, J.A. (1983). Synthesis of research on computer-based instruction. *Educational Leadership.* 44, 19-21.

Larson, D.E. (1981). The use of computer-assisted instruction to teach calculation and regulation of intravenous flow rates to baccalaureate nursing students. Doctoral Dissertation. Michigan State University, 1981. *Dissertation Abstracts International.* 42, 3459A.

McDowell, B.J., Nardini, D., Negley, S. and White, J. (1984). Evaluating clinical performance using simulated patients. *Journal of Nursing Education.* 12, 37-39.

Norman, G.R., and Painvin, C. (1987). Computer simulations in *Assessing Clinical Competence.* Edited by V.R. Neufeld and G.R. Norman. New York: Springer Publishing Company.

Conceptual Understanding of Software Packages

Danielle R. Bernstein
Kean College

Abstract

In a college level introductory microcomputer applications course taught to management science majors, word processing, spreadsheets, and database management systems are typically introduced sequentially. This is commonly a skill-based course in which software packages are taught procedurally (i.e., in a step-by-step manner). The course described in this project builds on the introductory course, but presents software applications as tools used to solve problems and borrows the ideas of synthesis skills from Elliott Soloway [Soloway, E. (1989). It's not too late to take a "shop class"; Learning synthesis skills through programming. National Education Computing Conference.].

The course simulates an office where students are computer power users—professionals who use a computer to solve problems and run several software applications regularly. Students select a system to design, implement, and demonstrate. Examples include a stock market tracking, an electronic "What's Doing on Campus," and a survey system. Several software packages are available to help them build the system. As students identify a need for a function (a need to search information or a way to keep random notes in a file as they work on a spreadsheet), they search out what they need to know about a possible software solution. Students work in small groups and act as colleagues sharing information and ideas. Class discussions emphasize the relationship and communality between the packages, choosing a package to do a particular job and how to approach learning a new software package.

The presentation will highlight the experiences of this approach, demonstrate examples of student projects, and discuss students' reactions to the course.

Using the Modular Approach to User Manuals in Teaching Documentation

W. Steve Anderson
University of Arkansas at Little Rock

Abstract

Many writing programs are strong on the notion that free writing, random association, and other intuitive techniques are important in the earliest phases of writing. These techniques may work for writing essays, stories, and other open-ended documents, but most students find that attempting to produce a manual by simply starting to draft does not result in an effective manual.

An effective user manual is more likely the result of an engineering process. Every feature of the program must be divided into smaller and smaller components until we can see the program application as a series of connected tasks. Only through this "engineering" process can we be certain that all the functional features inherent in the program have been accounted for. The first time I taught a course in user documentation, I tried to approach the writing process with a variety of drafting strategies, but the best results came from exercises based on the structured approach.

In the second offering of this course (Spring 1990), I will use the structured approach as the primary drafting strategy. My discussion will focus on the results of the modular approach and its potential application to other writing tasks.

Interactive High Speed Computer Classroom

Richard A. Alo' and Carol Jones Vobach
University of Houston—Downtown

Abstract

An interactive, high speed computer classroom has been developed at the University of Houston—Downtown under the joint funding of the National Science Foundation and the University of Houston—System. This innovative classroom has been functioning for 2 years and incorporates a state-of-the-art local area network of 17 IBM PS II Model 50 machines controlled by a PS II Model 60, which in turn is networked to the institution's VAX 8550. The initial design of the system incorporated curriculum adjustments and development for the originally intended audience of two junior level sequential statistics courses for business students satisfying a "short" calculus prerequisite. A motivating factor for developing such a classroom was to give mathematically shy business students (or others) a greater appreciation of statistical concepts and their applications to real world problems, more first-hand experience in dealing with popular IBM machines, and to increase interest in pursuing optional senior level mathematics courses in the area of decision sciences. Additional applications of the interactive classroom were directed toward a general computer literacy course, a Pascal course, and mathematics courses like numerical analysis. The pedagogical approach in the teaching delivery system was to incorporate a laboratory into the classroom environment, encouraging both an in-class team approach to problems and individual computer execution of manually solved problems (on the blackboard) with the immediacy of teacher/student emphasis on the analysis of output. The dynamics of this type of in-class situation require high speed desktop machines, individual storage, network accessibility, significant and rapidly retrieved overhead projected demonstrations (resolved through the use of a networked overhead viewgraph with one mega byte of memory storage), curriculum modifications, and resolution of certain implementation considerations and evaluation procedures (both for individual student testing and for determining the success of the program). These items are discussed in the presentation of the project along with the intricacies of the network and its development and the analysis of the motivating factors for establishing the classroom.

Computers and the Visual Artist

Marc J. Barr
Middle Tennessee State University

Abstract

For the past three years, I have been involved in teaching Fine Arts Foundation Design courses (2-Dimensional Design, 3-Dimensional Design, Color Theory, and Basic Drawing) at The Memphis College of Art and now at Middle Tennessee State University. I integrate the use of computers, with traditional materials, extensively into all of these courses.

In Basic Drawing, students can work directly with a digitizing tablet or manipulate existing images captured with a video camera or flatbed scanner. It is also a very useful way of demonstrating the concepts involved with perspective.

In 2-Dimensional Design, the students are able to work with the various elements of point, line, plane, and the manipulation of space, in a quick and creative manner.

In 3-Dimensional Design, I can easily demonstrate the principles of serial planes, modules, and structure of forms. The students can build and render their forms first as a wire frame model. They are then able to modify the surface, through the manipulation of color, surface texture, and lighting effects.

Color Theory has always involved a vary labor intensive trial and error mixing of palettes, in order to gain an understanding of hue, saturation, and value, before they can be applied to a design. With the aid of the computer, the student is able to create a variety of palettes and designs before rendering the final selection with gouache or acrylics. They are able to try many more variations and still get the experience of actually mixing pigments.

I also teach two upper divisional classes that involve the use of the computer as an "electronic sketchpad." In these classes, the students are able to combine all possible applications of the computer combined with traditional materials to produce finished drawings, paintings, and fabric designs.

Principles for Establishing a State Public Education Network

Connie Stout, Chair
Texas Education Agency
Austin, TX

Abstract

Computer networks have been used extensively in higher education and business settings for over two decades. Large numbers of these networks are now appearing in the K-12 educational settings. Many state departments of education have received mandates to establish networks that link all the school districts in their state. Although computer networks should be as accessible and usable as phone networks frequently that is not the case. Public school administrators may be reluctant to join any system until a standard has been established.

The plan will discuss some of the following generic principles:

- Architecture
- Interface
- Equity
- Applications
- Administrative
- Instructional

Panelists

Alex Belous
Arizona Department of Education
Phoenix, AZ

Judi Harris
University of Virginia
Charlottesville, VA

Paul Resta
University of New Mexico
Albuquerque, NM

Sponsor: ISTE

Computers as Learning Tools Across the Curriculum: Vanderbilt University's Experience with an Electronic Classroom

William F. Hogue
Vanderbilt University

Abstract

Vanderbilt University has equipped a classroom in a newly constructed building with 30 "fully-loaded" Macintosh IIx computers for students, one for the instructor, a local area network with connections to the campus backbone, a local filer server, and a variety of complementary audio-visual equipment—including full color projection, video and compact disk, video tape, "zoomable" overhead projection, and screen capture and control of each workstation.

In the first two semesters of operation economics, calculus, psychology, mechanical engineering, sociology, and molecular biology sections were taught in the classroom; in addition, a Spanish language laboratory has made use of the facility.

As with any new initiative, classroom management and use has not been without problems. However, responses from faculty and students have been overwhelmingly positive, with the vast majority planning to use the facility for teaching and learning again in the future. Preliminary findings suggest increased scope of understanding of subject matter among students, and an alteration of faculty pedagogical approaches at a fundamental level.

This presentation describes the design, installation, and operation of the classroom, including a rationale for hardware and software choices; student and faculty responses to hardware, software, and physical surrounding via surveys; a brief discussion of possible long term effects on learning at Vanderbilt and elsewhere; and potential follow-on activities and additional facilities based on what we have learned thus far. The emphasis will be on *showing* what has been done, rather than simply *telling* what has been done.

Teaching Analogical Reasoning through Guided Logo Programming

Neal Grandgenett
University of Nebraska at Omaha

Ann Thompson
Iowa State University

Abstract

The study investigated two effects of using guided Logo programming instruction to teach analogical reasoning within a classroom setting. The first effect investigated was the far transfer effect of such instruction on general analogical reasoning development, as measured by a test associated with general analogical reasoning. The second effect investigated was the near transfer of such instruction on a related computer programming skill—the reuse of subprocedures between programming problems. To provide general analogical reasoning training within a guided programming environment, the study incorporated Swan and Black's suggested pedagogical components for the effective transfer of cognitive skills from programming, and Sternberg's component processes of analogical reasoning. Far transfer results indicated significant interaction between a student's college year and the experimental treatment, with guided programming instruction facilitating the performance of college freshman, and hindering the performance of college juniors. Near transfer results indicated that the guided instruction did not significantly increase student reuse of subprocedures between programming problems.

Introduction

In 1933 John Dewey suggested that the major purpose of education was teaching people to think. He stated that "education...is vitally concerned with cultivating the attitude of reflective thinking, preserving it where it already exists, and changing looser methods of thought into stricter ones whenever possible" (p. 78). Such a goal would seem all the more relevant in today's complex world; and indeed, many educators are suggesting an increased emphasis on critical thinking for school curriculums (White, 1987).

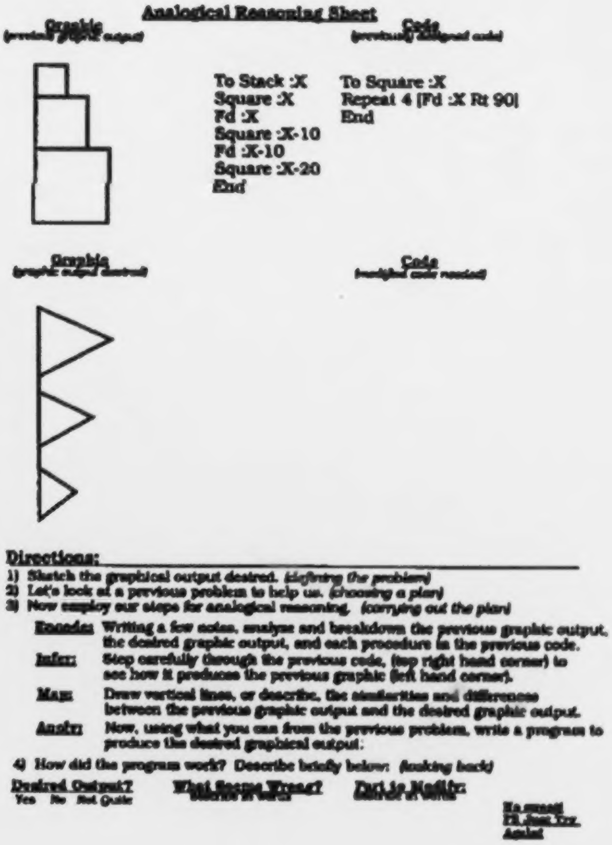
Analogical reasoning has been identified by numerous researchers as an exceptionally important cognitive skill in the general set of abilities that comprise critical thinking and problem solving (i.e., Sternberg, 1977b; Gick and Holyoak, 1980; & Halpern, 1987). Analogical reasoning is typically defined as the ability to utilize a well understood problem to provide insight and structure for a less understood problem (Gentner, 1982). For example, when a student is learning about the structure of an atom, he or she might assist understanding by referencing previous learning about the structure of the solar system. Such reasoning would seem to permeate everyday life, as previous experiences are used to understand current situations, and former problems are referenced to gain insight into new ones.

Can students be taught to be good analogical reasoners in light of the global nature of the skill? Although this question has been posed by researchers (Holyoak, 1984) few empirical studies have been attempted. Most of the research that does exist has investigated analogical reasoning within a controlled laboratory setting, such as the work by Sternberg, Ketron, and Powell (1982). Very few studies have investigated analogical reasoning training within the dynamic environment of the typical classroom.

General analogical reasoning training is not easy to incorporate into the classroom. Although students and teachers may often draw upon previous similar examples and problems in the analysis of new problems, such interactions are rarely seen by the students as incorporating a general and systematic thinking skill. The overall process of reasoning by analogy may be unapparent to the student due to their immersion in the details of the instruction. Classroom training focused on improving this reasoning ability may need to make such a process more explicit and apparent to the student. Instruction may need to be formally structured to make students aware of the general analogical reasoning process and to assist them in utilizing it effectively.

A direct attempt at the instruction of analogical reasoning in the language arts classroom was made by Alexander, White, Haensly, and Crimmins-Jeanes using 4th, 8th, and 10th graders (1987). By using the Sternberg componential model (1977a), analogical reasoning training was incorporated into an existing language arts curriculum. Using Sternberg's four components of encoding, infering, mapping, and applying, students gradually moved from working with concrete nonverbal analogies to more abstract verbal analogies. Incorporating the Woodcock Reading Mastery Test as an outcome measure, these researchers found a significant effect for their classroom analogical reasoning training. It was suggested by these researchers that analogical reasoning training using Sternberg's component processes appeared promising, but that further research, especially with different age groups and content areas, was needed to determine the effects of componential training in analogical reasoning.

A content area that seems particularly adaptable to classroom-based training in analogical reasoning is the area of computer programming. Programmers often need to draw systematically on prior programs in the development and construction of a new program, a process closely associated with general analogical reasoning (Pennington, 1982). The activity of computer programming itself has long been investigated as a potential rich environment for the exercise of critical thinking skills (i.e. Papert, 1980; Feurzig, Horowitz, and Nickerson, 1981), and some cognitive benefits from computer programming have been demonstrated in studies in this area (i.e. Clements and Gullo, 1984; Degelmen, Free, Scarlato, Blackburn, and Golden, 1986). However, only a few studies have specifically targeted analogical reasoning in the programming process (Mann, 1986; Clement, Kurland, Mawby, & Pea, 1986; Swan & Black, 1987). Such studies have investigated analogical reasoning as a general benefit to computer programming, but have not as yet explored the direct training of analogical reasoning by the use programming activities. This study focused more directly on analogical reasoning as a possible outcome to structured programming activities, and thus investigated the potential use of guided computer programming for the classroom training of analogical reasoning.



Analogical Reasoning Sheet

Graphic (previous graphic output)

Code (previously designed code)

```
To Stack :X
Square :X
Fd :X
Square :X-10
Fd :X-10
Square :X-20
End
```

```
To Square :X
Repeat 4 [Fd :X Rt 90]
End
```

Graphic (graphic output desired)

Code (modified code needed)

Directions:

1) Sketch the graphical output desired. *(defining the problem)*
2) Let's look at a previous problem to help us. *(choosing a plan)*
3) Now employ our steps for analogical reasoning. *(carrying out the plan)*

Encode: Writing a few notes, analyze and breakdown the previous graphic output, the desired graphic output, and each procedure in the previous code.

Infer: Step carefully through the previous code, (top right hand corner) to see how it produces the previous graphic (left hand corner).

Map: Draw vertical lines, or describe, the similarities and differences between the previous graphic output and the desired graphic output.

Apply: Now, using what you can from the previous problem, write a program to produce the desired graphical output:

4) How did the program work? Describe briefly below: *(looking back)*

Desired Output? Yes No Not Quite

What Seems Wrong? describe in words

Part to Modify: describe in words

[illegible]

Figure 1. Analogical Reasoning Structured Programming Worksheet

Method

The sample

The study was conducted by using 144 students enrolled in an introductory educational computing class at Iowa State University during the Fall 1988 semester. Most of the students were elementary education majors, although some secondary education and non-education majors made up approximately 34% of the group. A strong majority of the sample, 80%, had not taken a college computer programming course, and only 4% of the sample had taken more than 1 course. The sample was about 75% female, associated mostly with the elementary education majors, and 25% males, associated mostly with the other majors.

The treatment groups

The sample was randomly assigned to two specific treatment groups, one group experiencing Logo programming instruction focused on the development of general analogical reasoning, and one group experiencing more traditional exploratory Logo programming instruction. Both treatment groups were exposed to the same programming content, drawn from the textbook: *LogoWorks: Lessons in Logo* (Cory & Walker, 1985), with only the method delivering that content varied. Treatments consisted of approximately twelve hours of instruction each, spread out over a three week course unit. LogoWriter, a newer version of Logo with improved editor and graphic capabilities, was used as the programming language for the unit.

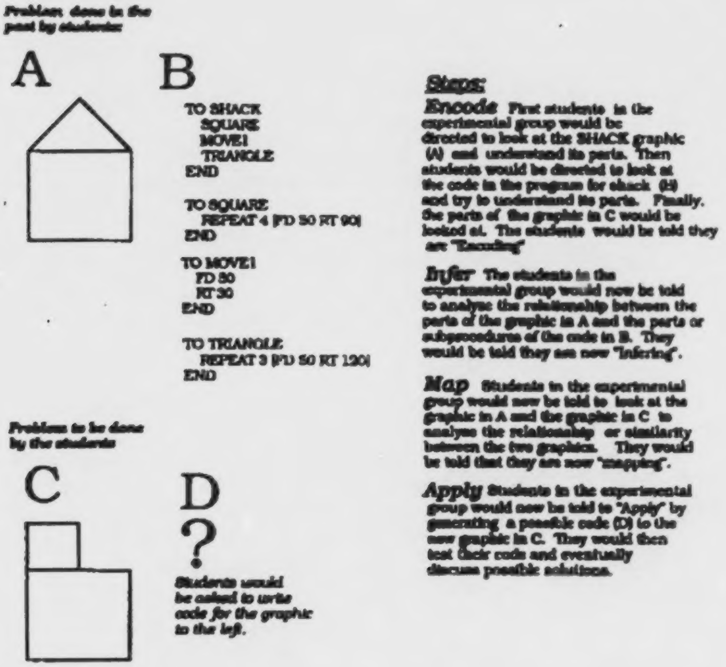
The treatment group which learned Logo programming by emphasizing general analogical reasoning procedures operated as the experimental treatment group. This instructional treatment consisted of a procedure similar to the Alexander, White, Haensly, & Crimmins-Jeanes (1987) procedure for the instruction of linguistic analogies. However, in this instruction the four parts of an A:B::C:D analogy were formed by pairing the output and corresponding computer code of a previously discussed program with the desired output and missing computer code of a new program. Students were taught with the assistance of structured programming worksheets incorporating this A:B::C:D analogy format (see Figure 1).

Since the purpose of the experimental treatment was to train students in analogical reasoning while they simultaneously learned computer programming, the instructional pedagogy of the experimental group emphasized this goal. Three instructional elements, drawn from the guided Logo research of Swan and Black (1987), were included in the instruction to help ensure that general analogical reasoning development was encouraged. These elements included an overall instructional focus on the analogical reasoning process, direct instruction of an analogical reasoning procedure, and a mediated approach to classroom discussion which continually placed examples and programs in a general analogical reasoning context. To directly instruct analogical reasoning in the experimental group, Sternberg's component processes of encoding, inferring, mapping, and applying (1977), were used as a framework for the programming instruction. Students used these component processes as thinking steps in drawing insight into the programming of some desired output by systematically referencing a previously completed problem. In the encoding step, students carefully listed all general characteristics of each of the parts of the A:B::C:D programming analogy. In the inferring stage, students traced through the code of the old program to try to review carefully how it produced its corresponding graphical output. In the mapping stage, students looked at the similarities and differences between the previous program output and the desired output. Finally in the applying stage, students attempted to use the knowledge gained in the first three steps, to now create programming code that produced the desired output (see Figure 2 for illustration). Each of the reasoning steps were accomplished in a paper and pencil format on the structured worksheets, with students later refining and testing their code on the computer.

Students in the control group were involved in Logo programming instruction using a more traditional discovery learning pedagogy. After a brief introduction of any new material, these students were instructed to immediately try to produce a specific graphical output by the writing of a corresponding Logo program. They were encouraged to look at former problems if necessary, but no step by step approach to do so was given. Instructional examples and content were the same as in the experimental group, but control group students did not use the initial analogical reasoning steps. Instead, the control group students were allowed to use their own strategies in planning their program. They had immediate access to the computer, and were encouraged to run and test their programming code a little at a time in an interactive fashion. This group also used instructional worksheets, however these were less structured and provided more of a typical Logo environment (see Figure 3). After the control group students worked 10-15 minutes on their own and completed the worksheet, the teacher would follow up the work session with a short class discussion on how best to solve the programming problem. Students were then allowed time to refine their programs if desired. This more exploratory method of learning programming more closely resembled the traditional discovery learning approach to Logo. Teacher assistance in the control group was given freely, usually by the use of the Socratic questioning typical of the Logo environment.

The Measures

The effects of the analogical reasoning training were investigated by the use of two measurement instruments which were assumed to represent the far and near transfer of training. Far transfer, or analogical reasoning development outside of the programming domain, was represented by the NonVerbal Battery of the Cognitive Ability Test (Thorndike & Hagen, 1986). Near transfer, or analogical reasoning development inside the programming domain, was


Problem done in the past by students:
A
B
Problem to be done by the students
C
D
?

```
TO SHACK
   SQUARE
   MOVE1
   TRIANGLE
END

TO SQUARE
   REPEAT 4 [FD 50 RT 90]
END

TO MOVE1
   FD 50
   RT 30
END

TO TRIANGLE
   REPEAT 3 [FD 50 RT 120]
END
```

Students would be asked to write code for the graphic to the left.

Steps:

Encode First students in the experimental group would be directed to look at the SHACK graphic (A) and understand its parts. Then students would be directed to look at the code in the program for shack (B) and try to understand its parts. Finally, the parts of the graphic in C would be looked at. The students would be told they are "Encoding"

Infer The students in the experimental group would now be told to analyze the relationship between the parts of the graphic in A and the parts or subprocedures of the code in B. They would be told they are now "Inferring".

Map Students in the experimental group would now be told to look at the graphic in A and the graphic in C to analyze the relationship or similarity between the two graphics. They would be told that they are now "mapping".

Apply Students in the experimental group would now be told to "Apply" by generating a possible code (D) to the new graphic in C. They would then test their code and eventually discuss possible solutions.

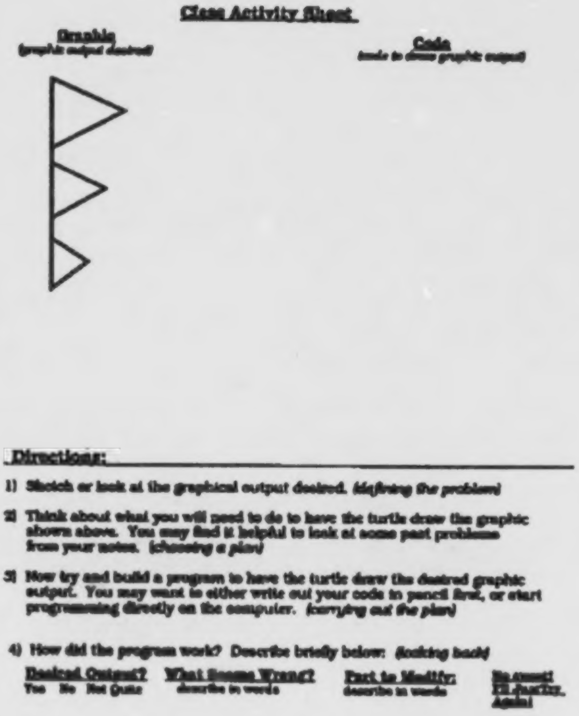
Figure 2. Sternberg's Component Processes Applied to Programming

investigated by the use of an instrument designed to provide an indication of the reuse of subprocedures between programming problems. In addition, a third measure, a LogoWriter basic comprehension test, was developed to compare groups on their basic understanding of the programming content.

The Cognitive Ability Test—NonVerbal Battery, used as a measure to represent the far transfer of analogical reasoning, is a commercially standardized instrument using full and partial analogies in various formats. It has been associated with prior work in analogical reasoning from a theoretical perspective (Mullholland, Pelegrino, and Glaser, 1980), and has also been used as an outcome measure investigating analogical reasoning in Logo programming (Mann, 1986). The non-verbal battery of the test is composed of three subtests which incorporate geometric non-linguistic analogies in various formats.

The reuse of subprocedures instrument, representing the near transfer of analogical reasoning, was a slightly modified version of a test developed at Bank Street College by Kurland, Clement, Mawby, & Pea (discussed in Kurland, Clement, Mawby, & Pea, 1987). The test was originally developed to investigate three aspects of program proficiency: reusability of code, flow of control, and program decomposition. The test was used by these researchers in a study which significantly correlated the reuse of subprocedures to success on a general analogical reasoning task (Clement, Kurland, Mawby, & Pea, 1986). In this study the test was modified to more directly focus on reusability of program code, or the reuse of subprocedures between programming problems. The modified test consisted of 5 graphical outputs sharing general characteristics to encourage and investigate the reuse of subprocedures between problems (see Figure 4). Scoring procedures were similar to the original procedures outlined by Kurland, Clement, Mawby, and Pea (1987), and produced a score which represented the total number of subprocedures reused between the five programming problems. The test was administered on-line, to give as realistic a programming environment as possible, and to aid in test scoring.

The basic LogoWriter comprehension test was a researcher developed, locally standardized instrument (KR20=.82), used to compare treatment groups on overall comprehension of the basic concepts of the LogoWriter language, which operated as instructional content for the study. It was decided that such an instrument was especially needed to help accurately interpret results on the reuse of subprocedures instrument; results that might have been heavily influenced by lower level differences in comprehension of the LogoWriter language. It has been included with this paper in the appendix (see Appendix A).



Class Activity Sheet

Graphic
(graphic output desired)

Code
(code to draw graphic output)

Directions:

1) Sketch or look at the graphical output desired. *(defining the problem)*

2) Think about what you will need to do to have the turtle draw the graphic shown above. You may find it helpful to look at some past problems from your notes. *(choosing a plan)*

3) Now try and build a program to have the turtle draw the desired graphic output. You may want to either write out your code in pencil first, or start programming directly on the computer. *(carrying out the plan)*

4) How did the program work? Describe briefly below: *(looking back)*

Desired Output?	What Seems Wrong?	Part to Modify:	Re-enter! I'll Just Try Again!
Yes No Not Quite	describe in words	describe in words	

Figure 3. Control Group Programming Worksheet

Procedures

Both treatment groups were given an initial 30 minute introduction to analogical reasoning and its component processes to ensure that any measured differences in the groups were primarily due to the programming instruction pedagogy, rather than an awareness of the analogical reasoning process. The experimental group then continued to use the analogical reasoning steps in their programming, while the control group assumed the more exploratory approach to the Logo programming.

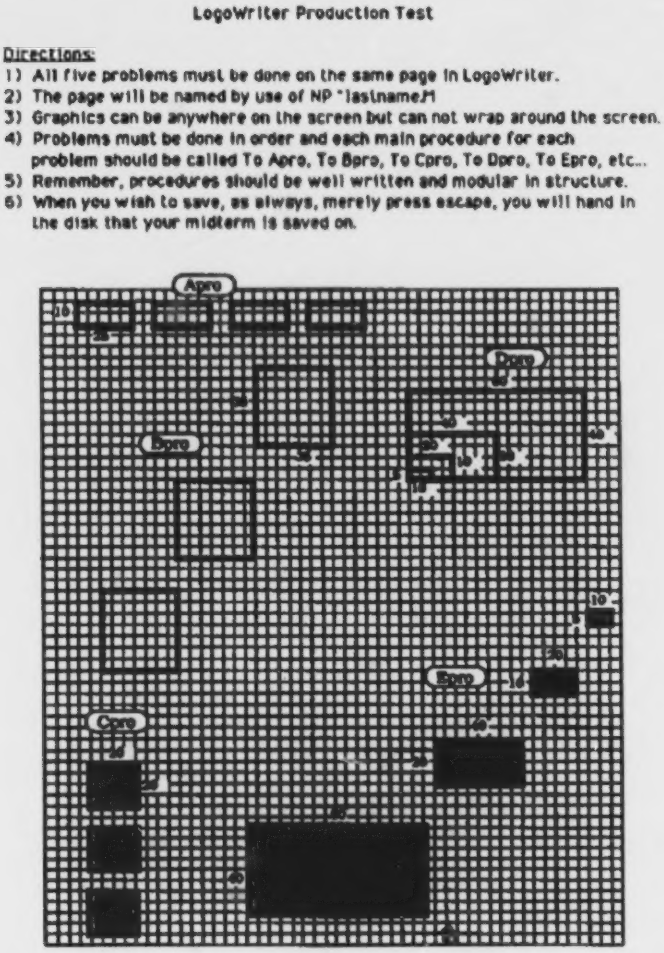
The experimental and control groups worked through the programming unit with the programming content and examples carefully matched between the two treatments. Instructional worksheets and guides were used to assist course instructors in following the particular pedagogy of each instructional treatment. Instructors were also randomly assigned to each course section, so that they each taught an approximately equal number of experimental and control classes. These class meetings were videotaped and reviewed to ensure that appropriate instructional pedagogy was carefully followed.

At the end of the three week unit, students were administered each of the three measurement instruments. The reuse of subprocedures programming test was used as part of a hands-on laboratory midterm for the course and the LogoWriter basic comprehension test was used as part of the written midterm. Although the Cognitive Ability Test could not be included as part of the regular course grade; it was however required that students take the test, and feedback of results were made available to the students.

Results

Initial Hypothesis Tests

Two hypotheses were tested in the study. The first hypothesis, representing far transfer, predicted that the guided Logo group would achieve a higher score on the Cognitive Ability Test—Nonverbal Battery than the more exploratory Logo group. The second hypothesis, representing near transfer, predicted that the guided Logo group would also reuse subprocedures more frequently on the constructed reuse of subprocedures test than the exploratory group. Each of these hypotheses were tested using a statistical *t*-test.



LogoWriter Production Test

Directions:
1) All five problems must be done on the same page in LogoWriter.
2) The page will be named by use of NP "lastnameM
3) Graphics can be anywhere on the screen but can not wrap around the screen.
4) Problems must be done in order and each main procedure for each problem should be called To Apro, To Bpro, To Cpro, To Dpro, To Epro, etc...
5) Remember, procedures should be well written and modular in structure.
6) When you wish to save, as always, merely press escape, you will hand in the disk that your midterm is saved on.

Figure 4. Reuse of Subprocedures Programming Test

The initial analysis of hypothesis one indicated there was no significant difference between group means for the CogAT-NB composite scores with $t = -0.28$, $p < .361$. Thus, initial results suggested no differential effects between groups on the study instrument used to represent far transfer of analogical reasoning.

For hypothesis two, group variances for the reuse of subprocedures scores were significantly different with an experimental group variance of 11.78 and a control group variance of 7.26. A Hartley's test for homogeneity of variances found this difference significant at $p < .043$, $F = 1.62$. A logarithmic transformation was then performed on the reuse of subprocedures data to stabilize variances. A *t*-test was performed on these transformed scores which indicated group means were not significantly different with $t = -0.45$, $p < .327$. Thus, similar to hypothesis one, the initial results for hypothesis two suggested no differential effects between groups on the study measure representing near transfer of analogical reasoning.

A statistical *t*-test was also performed on group means for scores from the LogoWriter basic comprehension test. This analysis indicated that group means were also not significantly different with $t = -0.83$, $p < .408$. Thus, as indicated by results from the LogoWriter basic comprehension test, both groups had achieved a relatively equal understanding of the instructional content.

Auxiliary Analyses

The study also explored the possible interactions of gender and college year with the instructional treatment. Both of these potential interactions were tested statistically by use of a multiple factor analysis of variance. The effect of gender was investigated for both the scores relating to performance on the CogAT-NB and scores on the reuse of subproceduresprogramminginstrument. For the CogAT-NB, neither gender/treatment interaction, nor gender alone was found to be a significant source of score variance with $p < .537$, and $p < .499$ respectively. Similar results were found with the reuse of subproceduresprogramminginstrument, with neither gender/treatment interaction or gender alone found to be a significant source of score variance at $p < .340$, and $p < .114$ respectively.

The effect of college year was also investigated for both outcome measures.

For the reuse of subprocedures programming instrument, neither year in college/treatment interaction, nor college year alone, was found to be a significant source of score variation with $p < .994$ and $p < .983$. Results differed for the cognitive ability test however. Although college year alone was not a significant source of variation, with $p < .246$, the interaction between year in college and instructional treatment was significant at $p < .033$ (see Table 1).

An interesting pattern was found for this interaction with college freshmen scoring highest when involved in the guided Logo instruction, college sophomores scoring about the same in both treatments, and college juniors and seniors scoring highest in the exploratory Logo instruction (see Figure 5). Further analysis of variance showed that only the differences in the freshman and junior subgroups were statistically significant, with $p < .047$, and $p < .029$ respectively.

Comparison of Mean CogAT - NonVerbal Battery Composite Scores for Both Treatment Groups by Year in College

A. Means and Counts

Treatment		**Experimental**[a] **35.42 (72)**[c]		**Control**[b] **35.07 (71)**
Year in College	**Freshmen** **36.14 (44)**	**Sophomore** **36.48 (31)**	**Junior** **32.54 (37)**	**Senior** **35.97 (31)**
Treatment By Year	**Exp. Fresh.** **38.96 (22)**	**Exp. Soph.** **36.40 (20)**	**Exp. Junior** **27.67 (12)**	**Exp. Senior** **35.17 (18)**
	Cont. Fresh. **33.32 (22)**	**Cont. Soph.** **36.64 (11)**	**Cont. Junior** **34.88 (25)**	**Cont. Senior** **37.08 (13)**

B. Analysis of Variance

Source of Variation	Sum of Squares	DF	Mean Square	F	Signif.[d] of F
Main Effects	**374.04**	**4**	**93.51**	**1.06**	**.38**
Treatment	**4.69**	**1**	**4.69**	**.05**	**.82**
Year in College	**369.76**	**3**	**123.25**	**1.40**	**.25**
Interaction					
Treat by Year	**794.59**	**3**	**264.86**	**3.01**	**.03***
Explained	**1168.63**	**7**	**166.95**	**1.89**	**.08**
Residual	**11897.80**	**135**	**88.13**		
Total	**13066.43**	**142**	**92.02**		

[a] **Logo systematically guided toward analogical reasoning.**

[b] **Logo taught in a traditional, exploratory approach.**

[c] **The numbers in parentheses denote sample size.**

[d] **The * denotes two-tailed significance at the .05 level.**

Table 1. Interaction of Treatment with College Year

Discussion

The results of the study suggest that analogical reasoning instruction through guided computer programming may be differentially effective for students at different grade levels and characteristics. In this study, the difference of only a few years in the college education level of the subjects was associated with a strong contrast between the training having a facilitating effect, or a hindering effect, for student performance on the study instrument representing far transfer of analogical reasoning.

One possible interpretation for the differing effects of the analogical reasoning training on the freshman and juniors, may be found in research discussions associated with ability/treatment interactions. Typically, students with lower general reasoning scores tend to work better with a more elaborated instructional treatment; one which provides a formal organizational structure for their learning. In contrast, students which score high on general reasoning instruments tend to prefer a less structured learning environment, allowing them more freedom in their reasoning tasks (Wittrock, 1974; Snow, 1980). It is interesting to consider whether such an interaction was found in this study. There is obviously a large mix of general characteristics that distinguish college freshmen from college juniors. Whether juniors actually differ significantly from freshman in specific characteristics such as general reasoning ability is difficult to say. However juniors do have the advantage of at least two years in formal instruction, and hold the inherent benefits of experiencing those two years. Freshman on the other hand, are relatively academically inexperienced, and are only beginning to experience learning at the college level. If general reasoning ability did play a role in this study, it seems likely that such general ability would be of a type that was "evolving" through college. This seems especially likely when considering that students were randomly assigned to study treatments, statistically equating groups in the more stable aspects of student general ability.

Further insight into the observed interaction in this study may be found from studies investigating differences in "crystallized" and "fluid" ability. Crystallized ability is associated with reasoning tasks drawing on verbal knowledge, reading comprehension, and prior achievement. Such ability tends to improve with experience. Fluid ability, in contrast, relates to

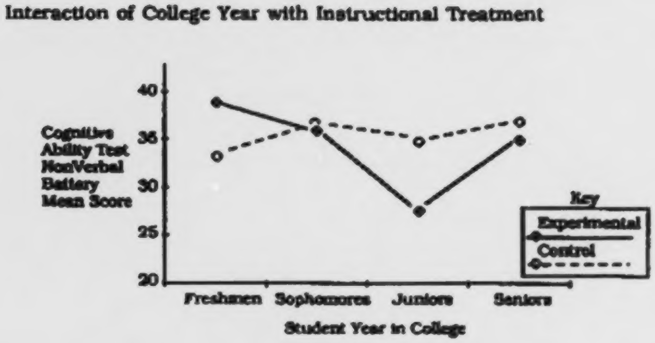


Figure 5. Interaction of College Year with Instructional Treatment

reasoning tasks which draw little on prior achievement, but encompass the ability to deal with new and relatively different processing tasks. Fluid ability usually encompasses the more stable aspects of general ability, and is somewhat resistant to change (Hart, 1986; Baltes & Schaie, 1982; Snow 1980).

The natural evolved difference between college juniors and college freshmen in their crystallized reasoning ability, therefore, may have been partially responsible for the observed difference in reactions to the experimental treatment. Juniors may have found the analogical reasoning instruction to be in conflict with the use of their current personal reasoning strategies, developed through general academic experience; encouraging a reduced performance on the Cognitive Ability Test. The reverse may have been true for college freshmen. Since their crystallized reasoning ability was less established, the treatment may have helped to facilitate their performance on the general analogical reasoning instrument by offering them a structured alternative to their less developed personal reasoning strategies.

It is interesting to note that a similar interaction between treatment and college year was not observed with the reuse of subprocedures instrument. Since this measure represented the near or same domain transfer of analogical reasoning, and the "guided" Logo treatment focused on far or different domain development, the potential for same domain effects for this skill may have been lessened. It may be that by focusing on the development of general analogical reasoning, little progress was made toward the development of more content specific analogical reasoning skills, such as the reuse of program subprocedures. Such a conjecture would seem in agreement with the far and near transfer ideas of Salomon and Perkins (1987).

Although more research is needed to determine the impact of possible interactions, classroom based analogical reasoning instruction through guided computer programming does seem to offer the potential of being useful for some students. Further research should focus on younger students, with associated lower levels of crystalized reasoning ability. It is possible that the interaction experienced in this study may not be experienced in studies involving younger children, since crystalized reasoning would be less developed. In any case, guided computer programming instruction does seem to offer a potentially useful avenue for current investigations of the potential for classroom-based instruction of analogical reasoning. Computer programming languages themselves, with their inherent and explicit problem solving structure, appear to offer a natural medium for investigating this important general reasoning process. A cognitive skill that is no doubt worthy of any efforts to help tomorrow's thinkers learn today.

Bibliography

Alexander, A. White, C. Haensly, P., & Crimmins-Jeanes, M. (1987). Analogy training: A study of the effects of verbal reasoning. *Journal of Educational Research*, *24*(3), 77-80.

Baltes, P. B., & Schaie, K. W. (1982). Aging and IQ--the myth of the twilight years. In S. H. Zarit (Ed.), Readings in aging and death: Contemporary perspectives. New York: Harper Row.

Clement, C., Kurland, D. Mawby, R., & Pea, R. (1986). Analogical reasoning and computer programming. *Journal of Educational Computing Research*, *2*(4), 473-485.

Clements, D. H., Gullo, D. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology*, *76*, 1050-1059.

Degelman, D., Free, J., Scarlato, M., Blackburn, J., & Golden, T. (1986). Concept learning in preschool children: Effects of a short-term Logo experience. *Journal of Educational Computing Research*, 2, 199-205.

Dewey, J. (1933). *How We Think: A Restatement of the Relation of Reflective Thinking to the Educative Process*, Lexington, Massachusetts: D.C. Heath.

Feurzig, W., Horowitz, P., & Nickerson, R. (1981). *Microcomputers in Education*. Cambridge Massachusetts: Bolt, Beranek, and Newman.

Gentner, D. (1982). Stucture mapping: A theoretical framework for analogy. *Cognitive Psychology*, *7*, 155-70.

Gick, M., & Holyoak, K. (1980). Analogical problem solving. *Cognitive Psychology*, *12*(3), 306-355.

Halpern, D. (1987). Analogies as a critical thinking skill. In D.E. Berger, K. Pezdek, W.P. Banks (Eds.) *Applications of Cognitive Psychology: Problem Solving, Education, and Computing*. Hillsdale, NJ: Erlbaum Publishers.

Hart, R. (1986). The effect of fluid ability, visual ability, and visual placement within the screen on a simple concept task. *Paper presented at AECT convention*, Las Vegas, NV. ERIC ED 267 774.

Holyoak, K. (1984). Analogical thinking and human intelligence. In R. J. Sternberg (Ed.), *Advances in the Psychology of Human Intelligence*. Hillsdale, NJ: Erlbaum Publishers.

Kurland, D., Clement, C., Mawby, R., & Pea, R. (1987). Mapping the cognitive demands of learning to program. In R. Pea and K. Sheingold (Eds.), *Mirrors of Minds*. Norword, New Jersey: Ablex Publishing, 103-127.

Mann, R. J. (1986). The effects of Logo computer programming on problem solving abilities of eighth grade students. *Dissertation Abstracts International*, 8619040.

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. New York, New York: Basic Books.

Pennington, N. (1982). Cognitive components of expertise in computer programming: A review of the literature. *Technical Report No. 46*. University of Michigan Center for Cognitive Science, Ann Arbor, Michigan.

Salomon, G., & Perkins, D. (1987). Transfer of cognitive skills from programming: When and how. *Journal of Educational Computing Research*, 3(2), 149-169.

Snow, R. E. (1980). Aptitude processes, aptitude, learning and instruction. In R. Snow, F. Montague, (Eds.), *Cognitive Process Analysis*. Hillsdale, NJ: Lawrence Erlbaum Publishers.

Sternberg, R. J. (1977a). Component processes in analogical reasoning. *Psychological Review, 84*, 353-378.

Sternberg, R. J. (1977b). *Intelligence, Information Processing and Analogical Reasoning: The Componential Analysis of Human Abilities*. New York, New York: John Wiley and Sons.

Sternberg, R. J., Ketron, J., & Powell, J. (1982). Componential approaches to the training of intelligent performance. In D. K. Detterman & R. J. Sternberg (Eds.), *How and How Much Can Intelligence Be Increased?*, Norwood, NJ: Abex Publishing, 155-172.

Swan, K., & Black, J. (1987). The cross-contextual transfer of problem solving skills. *CCT Report, 87*(3), Columbia University, New York.

Thorndike, R. L., & Hagen, E. (1986). *The Cognitive Ability Test*, Chicago Illinois: Riverside Publishing.

White, C. S., & Alexander, P. A. (1984). Teaching analogical reasoning processes. *Reading World, 24*(1), 38-42.

Wittrock, M. C. (1974). A generative model of mathematics learning. *Journal for Research in Mathematics Education, 5*, 181-196.

LogoWriter Basic Comprehension Test

Name________________________

Directions: Please read the following questions carefully and select the best answer for each question. In questions involving graphics, or sequences of specific commands, always assume that the turtle starts in the home position unless the question states otherwise.

1. Examine the following primitive command descriptions; which of the descriptions are incorrect?

Fd - moves the turtle forward a certain distance
Rt - turns the turtle to the right a certain number of degrees
Home - clears the screen and moves the turtle to the screens center facing up.
Fill - fills a graphic shape with a specific color
Pu - picks up the drawing pen of the turtle so that no line is drawn as the turtle moves

a. all of the descriptions are correct.
b. one of the descriptions is incorrect.
c. two of the descriptions are incorrect.
d. three descriptions are incorrect.
e. the descriptions are all basically correct, but the primitive commands must be typed in all capital letters for them to work.

2. In the LOGO programming Language, which of the following is not a primitive?

a. Cg
b. Fd
c. Seth
d. Fillit
e. Home

3. In Logo, the "primitive" commands are:

a. Useful procedures invented and defined by the user to perform some task, like moving the turtle forward or drawing a triangle.
b. Useful procedures that are already defined in the Logo language when it starts up.
c. The basic movement commands of FD, BK, RT, and LT, which are the only commands that actually move the turtle on the screen, and thus the only "primitive" commands.
d. The commands of PU, PD, PE, Home, HT, ST, and CG, which are the only commands that require no input numbers, thus they are the only "primitive" commands.
e. None of the above statements is correct.

4. Given the following sequence of primitive commands, and the information that the turtle is facing directly to the right of the screen, (►), before the commands are executed, which way does the turtle face after the commands are executed?

Fd 50
Rt 90
Fd 100
Rt 180
Bk 40
Lt 90

a. The turtle now faces to the bottom of the screen. ▼
b. The turtle now faces to the left of the screen. ◄
c. The turtle now faces to the top of the screen. ▲
d. The turtle still faces to the right of the screen. ►
e. It is impossible to tell without specific coordinates.

5. Which of the following sets of commands will position the turtle the greatest distance away from the home position? (assume that the turtle starts in the home position)

a.
Fd 100
Bk 100
Rt 90
Fd 100
Bk 40

b.
Fd 200
HT
Fd 100
Home
Fd 20

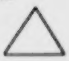
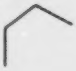
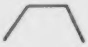
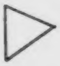
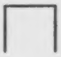
c.
Bk 100
Rt 90
Ht
Rt 90
Fd 70

d.
Fd 100
BK 200
Fd 25
Ht
Fd 50

e. It is impossible to tell without typing these commands into the computer.

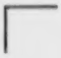
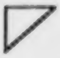
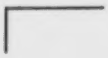
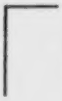
6. What will the following sequence of commands draw? (assume that the turtle starts in the home position)

Fd 50
RT 60
FD 50
RT 60
Fd 50
RT 60

a.

b.

c.

d.

e.


7. What will be drawn by the following sequence of commands? (assume that the turtle starts in the home position)

Fd 50
Rt 90
Fd 50
Home
Fd 50

a.

b.

c.

d.

e. None of These

8. Which of the following Repeat commands will not produce an error message when it is executed?.

a. Repeat [Fd 50 Bk 50 Rt 60]
b. Repeat Fd 50 [Rt 90]
c. Repeat 3 (Fd 50 Bk 50)
d. Repeat 4 [Pu Rt 90 Fd 50 Pd Bk 50]
e. All of the above statements will produce error messages.

9. Which of the choices below is the most efficient replacement for this set of commands to the right?

Setc 3
Fd 50
Rt 70
Fd 50
Rt 70
Fd 50
Rt 70
Fd 50
Rt 90

a.
Setc 3
Repeat 3 [Fd 50 Rt 70]
Fd 50
Rt 90

b.
Setc 3
Repeat 4 [Fd 50 Rt 70]
Rt 90

c.
Setc 3
Fd 200
Rt 210
Rt 90

d.
Repeat 3 [Setc 3 Fd 50 Rt 70]
Fd 50
Rt 90

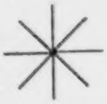
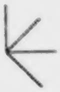
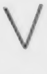
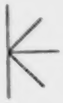
e. Repeat 3 [Setc 3 Fd 50 Rt 70]

10. In Logo, the Repeat command:

a. will make the turtle do something exactly twice. (for instance: Repeat Square draws two squares exactly the same).
b. provides the capability to simplify repeated sequences of commands into a single more efficient command.
c. must be used when drawing a square, triangle, rectangle, or circle.
d. will make the turtle do something over and over forever, until the programmer presses the "open-apple" and "S" keys.
e. none of the above are correct.

11. What shape would the following repeat command draw? (assume that the turtle starts in the home position) ▲

Repeat 5 [Fd 50 Bk 50 Rt 45]

a.

b.

c.

d.

e. None of These

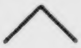
12. Which of the following procedures <u>will not</u> produce an error message when the procedure is executed?

```
a.  Vee        b.  To        c.  To Vee     d.  To Vee     e. all of these
    Lt 45          Vee           Lt 45          Lt 45         will  produce
    Bk 50          Lt 45         Bk 50          Bk 50         error messages
    Rt 90          Bk 50         Rt 90          Rt 90
    Fd 50          Rt 90         Fd 50          Fd 50
    End            Fd 50         Stop           To End
                   End
```

13. In LogoWriter, the term "Procedure" basically stands for:

a. the technique for drawing step by step pictures with a computer
b. a set of defined command steps to perform some task
c. the important problem solving steps of defining the problem, choosing a plan, carrying out the plan, and looking back at the solution.
d. all the important commands for using the editor, such as "open-apple-f"
e. none of the above

14. Which of the following procedures would correctly draw the figure shown below? (assume that the turtle starts in the home position)

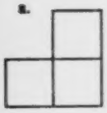


```
a.  To Peak     b.  To Peak     c.  To Peak     d.  To Peak     e.  None
    RT 45           Fd 50           RT 45           Rt 90           of
    Fd 50           RT 90           Fd 50           Fd 50           These
    RT 45           Fd 50           RT 90           Rt 45
    Fd 50           End             Fd 50           Fd 50
    End                             End             End
```

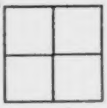
15. Given the Square procedure, what would be the graphical result of the following sequence of commands? (assume that the turtle starts in the home position)

```
Command Sequence:            To Square
Cg                           Repeat 4 [Fd 50 Rt 90]
Repeat 4 [Square Rt 90]      End
Fd 50
Square
```

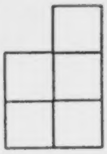
a.



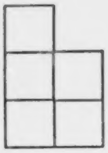
b.



c.



d.



e. None of These

16. When using the LogoWriter editor, it is important to:

a. press "open-apple-f" when entering the editor and "escape" when exiting the editor.
b. begin every student defined procedure with the word "To" and end every student defined procedure with the word "End".
c. begin a brand new page for each new procedure.
d. none of the above are correct.
e. all of the above are correct.

17. The following is an example of a program in LogoWriter:

```
To Blossom                   To Stem    To Flower   To Square
Repeat 10 [Square Rt 36]     Home       Stem        Repeat 4 [Fd 50 Rt 90]
End                          Fd 100     Blossom     End
                             End        End
```

Which of the following statements is true?

a. Blossom is the main calling procedure for this program.
b. Square is the main calling procedure for this program.
c. Flower is the main calling procedure for this program.
d. Stem and Blossom are both main calling procedures for this program.
e. There is no main calling procedure for this program.

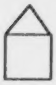
18. What is one of the reasons that a programmer might want to divide up a procedure into a calling procedure and various sub-procedures?

a. Because the LogoWriter editor only works with small procedures of no more then one screen long.
b. Because it is easier to analyze a problem, and program its solution, in parts.
c. Because sub-procedures like Square, Triangle, and Circle are already built into the Logo language, and these won't have to be created by the programmer.
d. Because in Logo there is no immediate mode, and the turtle can not execute a command unless it is written into a sub-procedure stored in the editor.
e. None of the above are true.

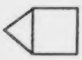
19. Given the following procedures in the workspace, what would be the graphic output when running the procedure "House"? (assume that the turtle starts from the home position)

```
To House     To Roof                   To Square
Square       Repeat 3[Fd 50 Rt 120]    Repeat 4[Fd 50 Rt 90]
Roof         End                       End
End
```

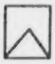
a.



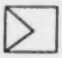
b.



c.



d.



e. None of these

20. Using the procedures of Frame, Wheel, & Handlebars, and assuming that each of these procedures draw only a specific shape, what is the super-procedure most likely needed for drawing a bicycle?

```
a. To Bicycle          b. To Bicycle    c. To Bicycle    d. To Bicycle    e. None
   Frame                  Frame            Frame            Frame            of
   Move1                  Wheel            Move             Move1            These
   Repeat 2 [Wheel]       Wheel            Wheel            Wheel
   Move2                  Handlebars       Move             Move2
   Handlebars             End              Wheel            Wheel
   End                                     Move             Move3
                                           Handlebars       Handlebars
                                           End              End
```

21. Looking at the following procedures, which of the statements below would be considered true?

```
To Mystery :X                   To Something :X :Y
Fd :X                           Fd :X
Rt :X + 90                      RT :Y
Repeat 100 [Fd :X Rt :X]        Repeat 100 [Fd :X Rt :Y]
End                             End
```

a. Both the Mystery procedure and the Something procedure use two variables.
b. The :X in the line "To Mystery :X", is unnecessary for input and could be removed.
c. The Mystery procedure could be executed by typing Mystery 47.
d. The Something procedure could be executed by typing Something 17.
e. More than one of these statements is true.

22. One of the reasons programmers may want to use variables in their procedures is because:

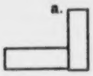
a. variables are needed in procedures to use the LogoWriter editor .
b. variable procedures are what make the graphics in LogoWriter colorful.
c. variables are needed for graphics, especially in drawing curved lines.
d. procedures using variables are more easily reused in other applications.
e. none of the above

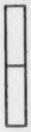
BEST COPY AVAILABLE

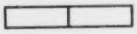
23. Using the following procedures, predict what happens when Train 20 50 is executed. (assume the turtle starts in the home position)

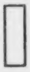
```
To Train :Width :Length
Rectangle :Width :Length
RT 90
FD :Length
LT 90
Rectangle :Width :Length
End
```

```
To Rectangle  :Width  :Length
Repeat 2[Fd :Width RT 90 FD :Length RT 90]
End
```

a.


b.


c.


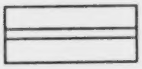
d.


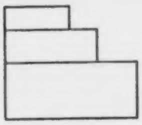
e. None of These

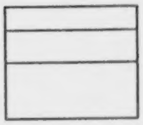
24. Which of the figures shown below will result from the execution of Stack 50?

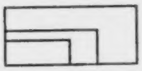
```
To Stack :X
Rectangle :X
Rectangle :X-20
Rectangle :X-30
End
```

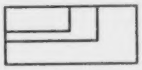
```
To Rectangle :X
Repeat 2 [Fd :X Rt 90  Fd :X * 2 Rt 90]
End
```

a.


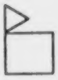
b.


c.


d.


e.


25. A student would like to design a LogoWriter program which will draw a triangle placed directly above a square, as in the picture on the right. She would like to have the side of the square and the side of the triangle to be different inputs. Which procedure below, would best fit her desire? (Square and Triangle are already in the workspace)



a.
```
To Fig :X :Y
Square :X
Fd :X
Triangle :Y
End
```

b.
```
To Fig :X
Square :X
Fd :X
Triangle :X
End
```

c.
```
To Fig :X :Y
Square :X
Fd :Y
Triangle :Y
End
```

d.
```
To Fig :X :Y
Square :X
Triangle :Y
End
```

e. None of these would be appropriate

26. Which of the following is an example of a procedure using recursion and a conditional statement to terminate it.?

a.
```
To Thing  :L
Fd :L
RT 5
Thing :L - 1
IF :L < 0 [Stop]
End
```

b.
```
To Thing :L
Repeat 4 [Fd :L Rt 5]
Fd :L
If :L < 0 [Stop]
End
```

c.
```
To Thing :L
For :L = 1 to 4
Fd :L
Rt 90
Next :L
End
```

d.
```
To Thing :L
Fd :L
Thing :L - 1
End
```

e. None of these

27. A "recursive" procedure in LogoWriter is a procedure that:

a. uses repeated curves within the graphical output.
b. is basically the same as a repeat statement but uses less commands.
c. calls itself as a sub-procedure.
d. calls more then two different sub-procedures.
e. all of the above are correct.

28. Looking at the following procedure, which of the statements listed below best describes the execution of the program?

```
To Lots
Repeat 2 [Fd 20 Rt 90 Fd 50 Rt 90]
Pu
Fd 20
Pd
Lots
End
```

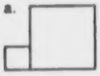
a. The procedure draws the same rectangle, in the same place, continually, until someone stops the program.
b. The procedure draws two rectangles, one above the other one.
c. The procedure draws one rectangle, moves forward, and then gives an error message.
d. The procedure continues to draw rectangles stacked above each other until the memory of the computer is filled up.
e. None of the statements above describe the execution.

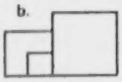
29. Given the procedures shown below, what figure would be drawn by Mystery 30? (assume that the turtle starts in the home position)

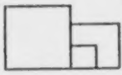
```
To Mystery :S
IF :S = 0 [Stop]
IF :S = 30 [RSquare :S]
IF :S < 20 [LSquare :S]
Mystery :S - 10
End
```

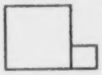
```
To Rsquare :S
Repeat 4[FD :S RT 90]
End
```

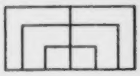
```
To Lsquare :S
Repeat 4[FD :S LT 90]
End
```

a.


b.


c.


d.


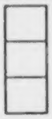
e.


30. In the following recursive procedure Blocks, what is the correct conditional statement to stop the procedure so that the output looks like the figure below when Blocks 3 is executed?

<u>Blocks Recursive Procedure</u>

```
(line 1)      To Blocks :x
(line 2)      Repeat 4 [Fd 50 Rt 90]
(line 3)      Fd 50
(line 4)      Blocks :x-1
(line 5)      End
```

<u>Desired Output</u>



a. Place the statement: "If :x < 0 [stop]" between lines 1 and 2.
b. Place the statement: "If :x = 0 [stop]" between lines 1 and 2.
c. Place the statement: "If :x = 0 [stop]" between lines 3 and 4.
d. Place the statement: "If :x = 0 [stop]" between lines 4 and 5.
e. None of the above

Functions, Transformations, and Mapping via Computer Graphics

Mary Dwyer Wolfe
Division of Natural Sciences and Mathematics
Macon College
College Station Drive
Macon, Georgia 31297

Tom Brieske
Department of Mathematics and Computer Science

Hiram Johnston
Department of Curriculum and Instruction
Georgia State University
University Plaza
Atlanta, Georgia 30303

Abstract

This paper describes the design of an interactive, computer graphics software package—Visually Interactive Computer Experiences in Mathematics [VICE Math]—that is designed to be used as a cognitive visualization tool to aid the learning of the concepts of function, transformation, mappings, and their compositions. Since it is a tool rather than a tutorial program, the software has no sequence of displays that instruct or question the user. Instead the software displays representations of the functions that are chosen by the user through the use of menus. This design allows the user to make and test conjectures about properties of functions. The software package has been successfully used in high school mathematics classrooms and in classes and workshops for inservice high school mathematics teachers.

Introduction and Rational

The concepts of function, transformation, and mapping are among the most important in mathematics. They are broad, deep, and powerful unifying forces for meaning. However, they are rarely perceived as such in elementary mathematics, particularly high school mathematics. To some extent this is inevitable since the concept of function cannot be understood all at once. Javier (1979) tells us that "understanding is a cumulative process mainly based upon the capacity of dealing with an 'ever-enriching' set of representations"(p. 67). The understanding of the concepts of function, transformation, or more generally, mapping comes only after gradual assimilation of multiple verbal and pictorial representations.

The medium of microcomputer graphics provides an exceptionally good learning environment for studying the concept of function. It is visual and dynamic, hence psychologically powerful. It is extraordinarily fast, thus allowing more time for examples. It is interactive, thus allowing students to choose their own examples for study.

Since the introduction of personal computers into schools and colleges, many software packages have been developed that can be used to graph functions in two and three dimensions. This paper describes an interactive computer graphics software package—Visually Interactive Computer Experiences in Mathematics [VICE Math]—that is a visualization tool that can aid in the learning of the mathematical concepts of functions, transformations, mappings, and their compositions. Its menu driven format allows the user to select mappings, compose them, apply them to various figures and to observe their properties. It also allows the user to make and test conjectures about mappings and their properties.

The VICE Math software, which was developed at Georgia State University, stresses the mapping interpretation of functions from the line to the line -- R to R functions, from the plane to the plane—R^2 to R^2 transformations, and from the line to the plane—R to R^2 functions. The VICE Math interpretation of each of the three types of mappings listed is given below with illustrated examples. We have used arrows in the figures in these examples to suggest the dynamic nature of the computer graphics output produced by the software.

Description of the VICE Math Software

Mappings from the Line to the Line—R to R Functions

The VICE Math software represents R to R functions in two ways by simultaneously drawing line-to-line mapping diagrams and Cartesian graphs for these functions. Figures 1,2 and 3 depict the function defined by the formula $F(x) = x + 1$. As shown in Figure 1, the user may select individual numbers in the domain of the function, such as -6 and 2, for the computer to map to their images, -5 and 3 respectively. Visually the correspondence between numbers in the domain and numbers in the range is dynamically represented by a sequence of dots starting in the domain and ending in the range. The dynamic process is suggested by the arrows in Figure 1.

In Figure 2 the "all points" option of the software is illustrated for the same function. When this option is selected, the computer maps several numbers of the domain to their images by sequences of dots as suggested by the arrows in Figure 2. In Figure 3, which is the final image produced by this menu option, the mapping *process* is illustrated by the mapping diagram on the left of Figure 3, and on the right, the *result* of the mapping process, that is the graph of the function, is displayed.

In Figures 4 and 5, the software illustrates the function defined by $F(x) = x^3 - x$. Again the arrows in the figures suggest the dynamic action of the computer which plots sequences of dots mapping numbers in the domain to their images in the range.

The software gives the user a wide choice of functions to illustrate. The menu of R to R functions includes linear, quadratic, cubic, rational, sine, cosine, exponential, logarithmic, and absolute value functions. The user selects the type of function from a menu and then is prompted to specify the appropriate parameters to define the function of choice.

Composition of R to R Functions

Line-to-line mapping diagrams are very effective representations of compositions of R to R functions. The VICE Math software uses mapping diagrams to illustrate the composition of two R to R functions. In Figure 6 on the left side, the mapping diagram for the function defined by $F(x) = 2x - 1$ is chained to the mapping diagram for the function defined by $G(x) = |x|$. The user may specify individual numbers in the domain of F, such as -2 and 2; then the software maps each number to its image in the range of F, -5 and 3 respectively. If these numbers are in the domain of G, the software maps them to their image under G which are 5 and 3 respectively. On the right side of Figure 6, the software represents the composition of F followed by G, $G \circ F$, by directly mapping -2 to 5 and 5 to 3. In general, for a given number x, the left side of the graphics screen displays the two stage process $G(F(x))$ and the right side displays the result of the composition, $G \circ F(x)$.

In Figure 7 the software illustrates the two stage process on the left side of the figure and the composed mapping on the right side of this figure for several points in the domain of $G \circ F$.

Transformations of the Plane—R^2 to R^2 Mappings

Transformations of the plane and their compositions are particularly suited for interpretation within the medium of computer graphics. In Figure 8, the computer display illustrates the geometric effect of the translation $T_{(3,2)}$ which maps each point of the plane to the point that is three units to the right and two units up. The software produces dynamic sequences of dots that represent this mapping process as it is applied to various points and figures in the plane. This dynamic mapping process is suggested by the arrows in Figure 8. The software allows the user to apply a particular transformation to a standard flag option as shown in the first quadrant of Figure 8, an arbitrary group of points as shown in the second quadrant, the graphs of linear, quadratic, cubic, sine, cosine, exponential, logarithmic, and absolute value functions as shown in the third quadrant, polygons as shown in the fourth quadrant, and figures the user can draw by moving a cursor.

In addition to translations, the user can choose from a menu to study a rotation about any point or a reflection over any line in the plane by applying these transformations to the various types of figures listed above. By applying a sequence of these transformations to a figure, the user can investigate compositions of these transformations. For example, in Figure 9 on the left the software has rotated the flag 90 degrees about the origin and then translated the image flag by the translation $T_{(5,-6)}$. On the right side of Figure 9, the software has computed the composition $T_{(5,-6)} \circ R_{90}$ to be the rotation of 90 degrees about the point (5.5, -0.5), $R_{90,(5.5,-0.5)}$. The textual description of this composition is displayed in a text window below the graphics window as the software visually illustrates the composed mapping with several dynamic sequences of rotation arcs drawn from the original flag to the resulting flag. In Figure 10, the reflection in the vertical axis, F_V, has been applied to the results in Figure 9 to form the new composition $F_V \circ T_{(5,-6)} \circ R_{90}$. On the left side of Figure 10, these three mappings are shown as separate components while on the right side of Figure 10, the software illustrates that this composition is reducible to the glide-reflection produced by the reflection, F_ℓ , in line ℓ whose equation is $y = x - .5$ followed by the translation $T_{(-5.5,-5.5)}$ which is parallel to line ℓ.

The software allows the user to include any number of translations, rotations and reflections in a composition. After the components are displayed on the left side of the screen, choosing the compose option from the menu causes the software to compute the composition, display the verbal description of the composition in the text window, and dynamically illustrate the composed mapping in the right half of the graphics window.

In addition to illustrating isometries of the plane as in Figures 8, 9, and 10, the software enables the user to view the images of various figures under the following mappings: dilations, shears, stretches, or more generally, the linear mapping $L(x,y) = (ax + by, cx + dy)$ where the user specifies the parameters a, b, c, and d. In Figure 11, for example, the software has shown that the dilation by in the origin, D_2, maps the small triangle to the large triangle in the first quadrant. This dilation is followed by a rotation, R_{180}, about the origin. In the right side of Figure 11, the display shows that the composition, $R_{180} \circ D_2$, is the dilation of -2 in the origin, D_{-2}.

In Figure 12, the linear mapping defined by the formula $L(x,y) = (x, -4 + y)$ or $(x,y) \to (x, -4x + y)$, which is a shear, has been applied to the graph C of the cubic function defined by $y = x^3$. The display shows that the image C′ of C is the graph of another cubic that has the equation $y = x^3 - 4x$. In Figure 13, the shear defined by $(x,y) \to (x, x + y)$ has been applied to the graph S of the sine function defined by $y = \sin x$. The display shows that the image of the sine curve S is the curve S′ that has the equation $y = x + \text{sine } x$.

Linear Mappings Defined by Matrices

The VICE Math software allows the user to define a linear mapping of the plane by specifying a 2X2 matrix of the mapping of the form:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

In Figure 14, the geometric effect of the linear mapping defined by the following matrix is illustrated:

$$\begin{pmatrix} 1.5 & 5 \\ .5 & 1.5 \end{pmatrix}$$

In this figure, $E_1 = (1,0)$, $E_2 = (0,1)$, $L(E_1) = (1.5,.5)$, and $L(E_2) = (.5,1.5)$. After the computer plots each point of the domain on the left side of the display, it them plots the corresponding image point in the range on the right side. The origin of the domain and its image are displayed first, the E_1 and its image, $L(E_1)$, then E_2 and its image, $L(E_2)$, then equally spaced points on horizontal lines and their images, and finally, a flag and its image. The following statements appear in the text window below the graphics:

> The mapping is one-to-one and onto.
>
> It maps unit squares in the domain
>
> to parallelograms in the range that
>
> have areas of .75.
>
> The kernel of the mapping is the origin.

If the linear mapping chosen by the user is not one-to-one and onto, for example the mapping defined by

$$\begin{pmatrix} 2 & 1 \\ 2 & 1 \end{pmatrix}$$

the software does the following. The domain and range are illustrated as shown in Figure 15. The software describes the range as all multiples of (2,2); then the graphics window is cleared and the origins are drawn again followed by the graph of the kernel of the mapping as shown in Figure 16. The text window indicates that the kernel of this mapping is all multiples of (-1,2). Next, selected points in the range are plotted along with their pre-images in the domain producing the family of lines parallel to the kernel shown in Figure 17.

Mappings from the Line to the Plane—R to R^2 Mappings

Mappings from R to R^2 defined by formulas of the form $t \to (f(t),g(t))$, where t represents time and f and g are chosen from a menu of standard R to R functions, model the motion of a particle moving in the plane. The VICE Math software dynamically represents this motion by presenting domain and range of such a function side by side.

In Figure 18, the mapping defined by the formula $t \to (4\cos(t+1), 2\sin 2t)$ is illustrated. On the left side of Figure 18, an increasing sequence of time values is represented by the arrow. As time values in the domain are plotted on the display screen, the corresponding image points in the range are also plotted on the right side of the display, thus suggesting the motion of a particle. The software plots the image of each value of t, and then it retraces the path of the motion with a continuous curve as shown in Figure 19.

R^2 to R^2 Non-Linear Mappings

The VICE Math software also allows the user to view mappings from the plane to the plane defined by specifying the formula $(x,y) \to (ax^2 + by^2 + cx + dy + exy + f, a'x^2 + b'y^2 + c'x + d'y + e'xy + f')$. In Figures 20 and 21, the mapping defined by $(x,y) \to (.25x^2-.25y^2,.5xy)$ is illustrated. The software plots a sequence of horizontal lines in the domain and their corresponding parabolic images in the range as shown in Figure 20. It then plots a sequence of vertical lines in the domain and their corresponding parabolic images in the range as shown in Figure 21. This figure shows how the mapping effects the standard coordinate grid in the domain. The user may also choose from a menu the option that plots families of diagonal lines parallel to $y = x$ and $y = -x$ along with the corresponding images in the range.

Summary and Conclusions

The VICE Math software was designed as a powerful cognitive visualization tool to aid in the study of the general concept of function. This menu driven tool does not tutor, but rather, it allows the user to explore important components of the general concept of function, either independently or as directed by a teacher. The VICE Math software has the capability to pictorially represent functions in the different contexts of R to R, R^2 to R^2, and R to R^2 mappings, thus promoting a more inclusive and powerful abstraction to the concept of function as well as a contrast between the ideas of *function as a process* as exemplified by the dotted lines and arcs from pre-images to images in the contexts of R to R and R^2 to R^2 mappings, and *function as an object* as exemplified by applying R^2 to R^2

transformations to the graphs of R to R functions. The menu driven format allows the user to select a large variety of mappings in each context, R to R, R^2 to R^2, and R to R^2 mappings, by specifying parameters. Finally, the software allows for composition of two R to R mappings and any number of R^2 to R2 mappings.

The VICE Math software can be appropriately used in high school mathematics classes, college mathematics classes, as well as in classes or workshops designed for in-service high school mathematics teachers. Also, it has been used as the focus of a very successful experiment in teaching high school seniors in a pre-calculus course with a strong emphasis on functions and transformations. The software is currently being used as an important part of an ongoing revision of a high school's mathematics curriculum that will place functions, mappings, and transformations at its center.

Obtaining Copies of the VICE Math Software

There are two versions of the VICE Math software. The Apple version was written in Apple Pascal 1.3. The self-booting disk will run on any Apple II series computer with 128K and one 5 1/4" disk drive. The IBM version of the software, which was written in Turbo Pascal 4.0, will run on any MS DOS machine with at least 256K, CGA, EGA, VGA, or MCGA graphics. To obtain a free copy of the VICE Math software, send a disk to Tom Brieske at the Georgia State University address listed above. Specify whether you want the Apple or the IBM version of the software. Permission to copy the disk for classroom use will be granted upon written request.

Bibliography

Bork, Alfred (1985). *Personal Computers for Education*. Englewood Cliffs: Prentice-Hall.

Churchhouse, R.F., et al. (Eds.). (1986). *The Influence of Computers and Informatics on Mathematics and Teaching*. Cambridge: Cambridge University Press.

Fey, James T., et al. (Eds.). (1984). *Computing and Mathematics: The Impact on Secondary Curricula*. Restin: National Council of Teachers of Mathematics.

Hansen, Viggo P. and Zweng, Marilyn J. (Eds.). (1984). *Computing and Mathematics Education*, 1984 Yearbook. Restin: National Council of Teachers and Mathematics.

Heid, M. Katherine. (1988, January). Resequencing Skills and Concepts in Applied Calculus Using the Computer as a Tool. *Journal for Research in Mathematics Education, 19*, 3-25.

Hearn, Donald & Baker, M. Pauline. (1986). *Computer Graphics*. Englewood Cliffs: Prentice-Hall.

Javier, Claude. (1987). *Problems of Represention in the Teaching and Learning of Mathematics*. Hilesdale: Lawrence Erlbaum Associates.

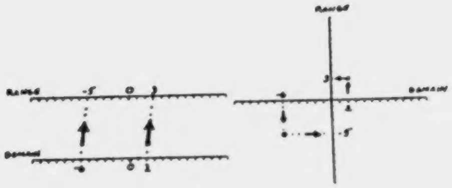
Kelman, Peter et.al. (1983). *Computers in Teaching Mathematics*. New York: Addison Wesley.

Oldknow, Adrian & Smith, Derek. (1983). *Learning Mathematics with Micros*. New York: Halsted Press.

Papert, Seymour. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books.

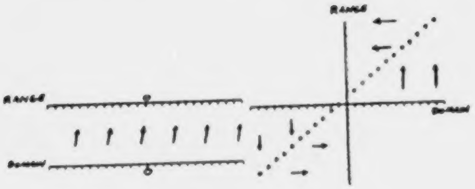
Thompson, Patrick W. (1984). *Motions: A Microworld for Investigating Motion Geometry*. Cosine, Inc.

Walker, Decker F. & Hess, Robert D. (1984). *Instructional Software: Principles and Perspectives for Design and Use*. Belmont: Wadworth.


RANGE
DOMAIN
RANGE
DOMAIN
0
0

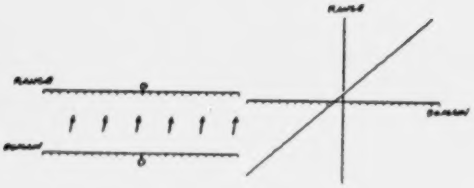
$f(x) = x+1$

Figure 1


RANGE
DOMAIN
RANGE
DOMAIN
0
0

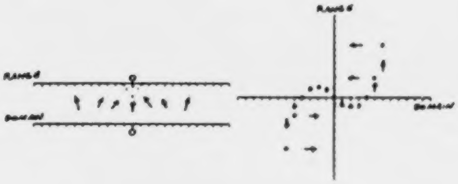
$f(x)=x+1$

Figure 2


RANGE
DOMAIN
RANGE
DOMAIN
0
0

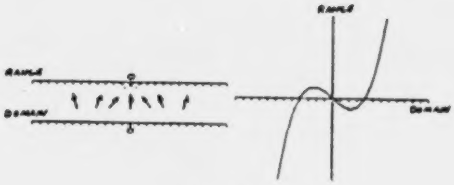
$f(x)=x+1$

Figure 3


RANGE
DOMAIN
RANGE
DOMAIN
0
0

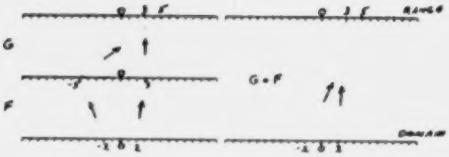
$f(x) = x^3 - x$

Figure 4


RANGE
DOMAIN
RANGE
DOMAIN
0
0

$f(x) = x^3 - x$

Figure 5


RANGE
G
F
G∘F
DOMAIN
0
0

$F(x) = 2x - 1$
$G(x) = |x|$

Figure 6

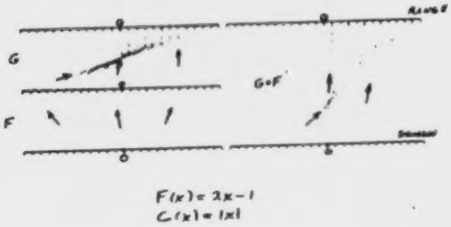

RANGE
G
G∘F
F
DOMAIN
0
0
F(x) = 2x - 1
G(x) = |x|

Figure 7

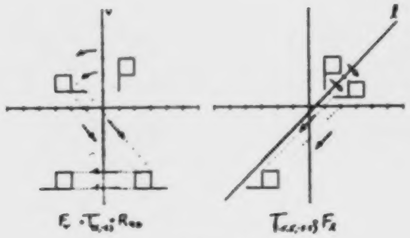


Figure 10

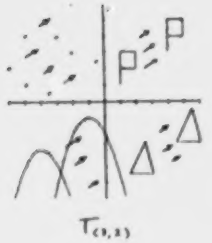


Figure 8

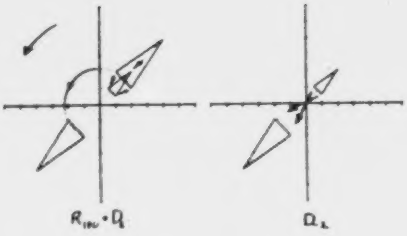


Figure 11

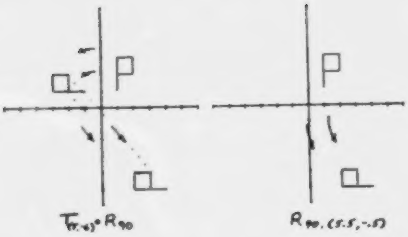

T ∘ R90
R90, (5.5, -.5)

Figure 9

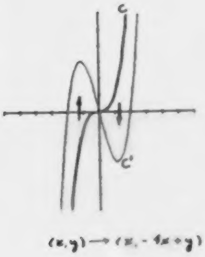
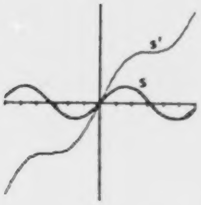


Figure 12



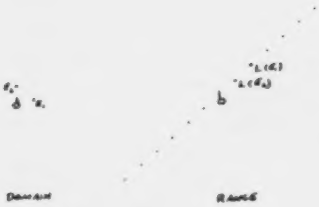
$(x,y) \to (x, x+y)$

Figure 13

DOMAIN
RANGE

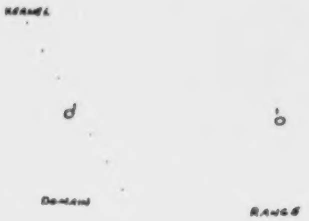
$\begin{pmatrix}1.5 & .5\\ .5 & 1.5\end{pmatrix}$

Figure 14


b
DOMAIN
RANGE

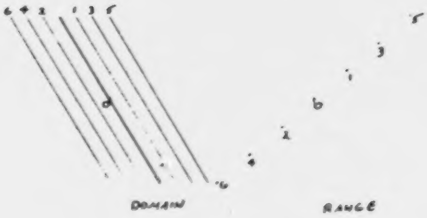
$\begin{pmatrix}2 & 1\\ 2 & 1\end{pmatrix}$

Figure 15


KERNEL
d
DOMAIN
RANGE

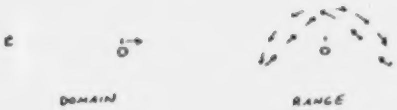
$\begin{pmatrix}2 & 1\\ 2 & 1\end{pmatrix}$

Figure 16


d
b
DOMAIN
RANGE

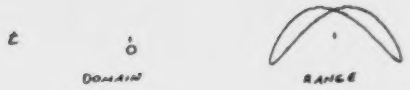
$\begin{pmatrix}2 & 1\\ 2 & 1\end{pmatrix}$

Figure 17


t
DOMAIN
RANGE

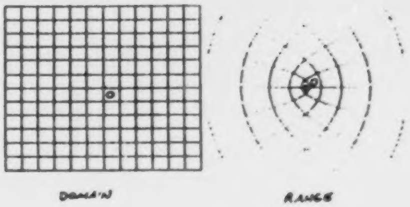
$t \to (4\cos(t+\pi), 2\sin 2t)$

Figure 18


t
0
DOMAIN
RANGE

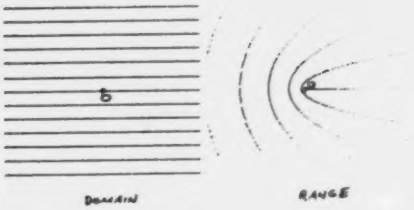
$t \rightarrow (4\cos(t+1), 2\sin 2t)$

Figure 19


DOMAIN
RANGE

$(x,y) \rightarrow (.25x^2-.25y^2, .5xy)$

Figure 21


DOMAIN
RANGE

$(x,y) \longrightarrow (.25x^2-.25y^2, .5xy)$

Figure 20

Characteristics of a Logo Programming Culture

James Dunne
Long Island University/C. W. Post Campus
Department of Educational Technology
Brookville, New York 11548
(516) 299-2147
EDT_DUNNE@LIUVAX

Abstract

Computer programming in general, and the Logo language in particular have long been advocated as activities that will lead to the development of higher order thinking skills. However, there is still controversy and doubt as to whether Logo delivers on this promise. Research on Logo has primarily focused on the *new technology* without adequately dealing with all of the other elements that make up the total learning culture. This paper focuses on the characteristics of a learning/programming environment or culture that may be necessary to help students develop higher order thinking skills through the process of programming.

Defining Logo

The implementation of a Logo learning culture is a complex innovation that encompasses physical, social, pedagogical, and curricular change. Along with progressive education, open classrooms, team teaching, and a host of other complex educational innovations, the developers of Logo have not been able to effectively communicate to the schools exactly what it is all about and how to use it. This lack of a commonly accepted definition has led, in many cases, to implementations with unclear or contradictory goals and fuzzy methodologies. Interpretations range from a delivery system requiring a hands-off approach to a lock step progression through the syntax and semantics of the language. Regardless of the interpretation, there is almost always an assumption that higher order skills will result. Usually, the result is disappointing.

When interpreted as being the focus of a complex learning environment or culture, Logo can be defined as a high level computer programming language that has its roots in the constructivist theories of Jean Piaget and in the need to better understand our own thought processes taken from field of Artificial Intelligence. It was designed to provide children with an environment rich in interesting things to do and to think about. In the process of learning and using Logo, children are thought to develop higher order logical/mathematical skills that are applicable to a much broader range of tasks than just programming. These skills are developed in the course of a child's intrinisically motivated explorations of Logo microworlds. These explorations require the child/programmer to engage in self-reflective behaviors that will lead to a better understanding of, and control over, his or her own thought processes.

Research on Logo and Logo-like programming environments has, for the most part, focused on the effectiveness of programming as a vehicle for the development of higher order thinking skills. Yet this research has been limited by Seymour Papert's notion of *technocentricity* (Papert, 1985). Most studies have focused on the "new technology" without adequately dealing with all of the other elements that make up the total learning culture. Given the logistical nightmare of attending to all of the variables in a complex learning environment, it seems unlikely that the question, *Does Logo work?*, will ever be definitely answered through experimental research. Nevertheless, the research provides some insight into what the characteristics of a successful programming environment or culture might be.

Characteristics of a Programming Culture

Schools are (or at least should be) learning cultures, constantly in the process of creating, adapting, and refining environments in order to help their students acquire knowledge and skills. Creating a learning culture centered on programming computers, and the problem solving that involves, is a new and unfamiliar task for most schools. Recent literature dealing with programming, Logo, and cognition indicates a need to focus on *both* the environment and the teaching methods in a programming culture. This paper will focus on the environment and the characteristics of a learning/programming culture that may be necessary to help students develop higher order thinking skills through the process of programming.

Adequate Resources and Sufficient Time

Most programming instruction currently in place in elementary through high schools suffers from a very high student to computer ratio and relatively little time spent programming (Becker, 1987; Dalbey & Linn, 1986; Linn, 1985; Wright, Melmed, & Farris, 1984). Short term exposure to Logo will not ensure even the most fundamental mastery of programming skills. Extended projects and long term use are necessary for students to acquire the general programming skills that are a prerequisite to using the higher order thinking skills that could transfer beyond the context of programming (Clements, 1985; Dalbey and Linn, 1986; Krasnor and Mitterer, 1984; Mawby, 1984). Hoyles

(1985) describes the importance of extended projects in a Logo environment as "a powerful means of moving students away from routine exercises and non-purposeful tasks ... [and] ... to set up 'situations to investigate' rather than 'problems to solve' " (p. 25). These situations to investigate are seen as ways to increase student motivation and self-esteem and to encourage collaboration. They are also necessary for students to build the extensive and rich history of experiences needed to develop and apply higher order skills.

All this will require considerably more hands-on time at the computer than students currently receive. Papert has often used the metaphor of the computer as a pencil to illustrate this need. As he sees it, before students will begin to receive any real benefit from using computers they will need frequent and easy access to them. In the same way that they now have access to pencils when they need to record information on paper, they will need access to computers whenever they find a need to explore an idea, solve a problem, access information, or accomplish a task.

Curriculum

Programming as it is currently being taught is usually limited to an introduction to the features of the language being used (Dalbey & Linn, 1986; Linn 1985; Wright, Melmed, & Farris, 1984). Bork, Pomicter, Peck, and Velso (1986) claim that the major problem with the teaching of programming is the enormous emphasis usually placed on grammatical detail. This emphasis is rooted in the necessity of understanding the syntax and semantics of a language before the learner can effectively use the language as a medium of expression. It is also rooted in the fact that it is relatively easy to construct a curriculum around the basic commands and rules of a language, while it is a much more difficult task to teach programming as a dynamic process. This is especially true if the teacher has only a rudimentary knowledge of programming.

Soloway (1985) found that most introductory programming textbooks stress the syntax and semantics of the language. While the textbooks sometimes attempt to teach problem solving in the context of teaching programming, he found that they don't typically say enough explicit about problem solving. Yet, Mandinach (1986) found that well developed curricula and curriculum materials were major factors in helping students develop and use higher order skills through programming. If the goal of teaching programming is to provide students with a generic set of skills that can transfer beyond the context of programming, then we need to move beyond an essentially rote approach to programming instruction.

Both a programming/problem solving curriculum and associated materials that: 1) emphasize the process of programming and 2) provide models for problem solving and reflective thinking need to be developed. They also need to illustrate how the strategies used to solve programming problems can be applied in other content areas, and in a variety of situations. This is especially important since programming (at least currently) is unfamiliar, and often threatening, to teachers (Bork *et al.*, 1986; Michayluk, 1986; Psotka, 1985).

Specifically, the following topics need to be addressed when designing a programming/problem solving curriculum:

Language Features

Clearly, students need a basic mastery of the syntax and semantics of a programming language before they can use it as a creative and exploratory tool. But, knowledge of the commands in a programming language and how they can be used is only a prerequisite to programming. Logo was designed to have a "low threshold" that would not require users to master most of the language features before they could accomplish something of interest and meaning to them. Mastery of a few basic commands facilitates an early entry to the process of programming. This also makes it easier for the teacher to introduce, and for students to learn, language features in context. The commands and constructs of the language need to be introduced, reintroduced, and applied over time in a variety of contexts and in varying combinations. It is important for educators to realize that knowledge of language features means that students are ready to reap the cognitive benefits of programming, not that it's time to move on to something else.

Problem Solving

Problem solving ability is not an automatic by-product of the programming process. Students need to be exposed to, and use, a variety of problem solving strategies applicable to their programming projects (Bork *et al.*, 1986; Clements, 1986; Dalbey, Tourniaire, & Linn, 1986; Perkins & Martin, 1986). Various models of these strategies being used in situations that are of meaning and interest to students are necessary to make them part of the students' problem solving repertoire (Dalbey, Tourniaire, & Linn, 1986; Weir, 1987). These models should be found in the curriculum materials that they use, and by observing real programmers (their teachers) solve real programming problems. In addition, Clements (1985) emphasizes the need to make students aware of how the problem solving strategies they use in programming can be applied to other classroom tasks. Without directly making these connections, the ultimate goal of having these strategies transfer beyond the context of programming may never be achieved.

The problem solving aspect of a programming curriculum needs to deal with: (a) the *powerful ideas* that are part of the programming language; (b) the *problem solving strategies* (cognitive and metacognitive skills) that are part of the programming process, and (c) the skills that help students develop effective *management strategies* for the knowledge they have.

Powerful Ideas. In Logo, the concept of Powerful Ideas refers to general heuristics that make it easier to rganize and debug programs and projects. A heuristic can be loosely defined as a generalizable approach to solving a certain type of problem. It can be thought of as an educated guess or trick of the trade. In order to reach a level of expertise in any field, it is often necessary to apply a variety of heuristic strategies when confronted with a new or unique problem. When people have a bag of (heuristic) tricks at their disposal they are better able to find something familiar in new a situation and adapt as a situation changes. Programming constructs such as iteration or branching are powerful ideas in themselves and can be applied in many contexts other than programming that require repetition and decision making. In addition, programming solutions such as algorithms become powerful ideas when they are used in a variety of projects or programs. For example, Logo commands that result in the turtle drawing a circle can be adapted (by using different inputs) to produce circle-like objects such as the sun, a tire, a balloon, a doorknob, or any polygon of any size. While most powerful ideas (heuristic strategies) evolve and are learned during the process of solving problems, the acquisition of these strategies can be expedited and nurtured. A programming curriculum needs to expose students to the powerful ideas that are part of the language. Through use and practice the students appropriate them and use them as models that enable them to discover or create their own powerful ideas.

Problem Solving Strategies. In addition to the powerful ideas that can be derived directly from the programming language, students need to be exposed to a more generic set of problem solving strategies that can be applied to their programming projects. These strategies involve the application of the cognitive and metacognitive skills that are at the heart of the programming/problem solving process. In Polya's (1957) classic model of problem solving these strategies are described as understanding the problem, devising a plan to solve the problem, carrying out the plan, and looking back to evaluate the solution. In programming these strategies translate to understanding the programming task, devising a plan—usually by breaking the problem/project down into more "doable" subproblems, looking for related problems or known solutions that apply, writing (programming) and testing (debugging) the code, recombining the subproblems to reach the final solution, and evaluating the completed project. Embedded in these strategies are many subskills such as trial and error, comparison, analogy, and working backwards. Again, there is no evidence that the acquisition of these skills is a natural by-product of programming. Left to their own devices children tend to rely on the relatively low level (inefficient and not always effective) skill of trial and error (Clements, 1985). A programming curriculum must provide students with models of these strategies.

Management Strategies. Introducing students to the powerful ideas and problem solving strategies involved in programming can't be done in a rigid linear sequence. Students need to build a backlog of programming/problem solving experiences before they can effectively use these strategies (Soloway, 1985; Dalbey and Linn, 1986; Perkins and Martin, 1986). Lack of experience usually causes two types of problems for beginning programmers. The first is what can be described as fragile knowledge. This occurs when the student either doesn't have the knowledge necessary to solve the problem, doesn't have all the knowledge necessary to solve the problem, or is using his or her knowledge in the wrong way. The second problem has been described by Collins, Brown, and Newman (1988) as a lack of control strategies. In their view, "as students acquire more and more heuristics and strategies for solving problems, they encounter a new management or control problem: how to select among the various possible problem solving strategies, how to decide when to change strategies, and so on" (p. 20).

A programming and problem solving curriculum needs to be flexible enough to introduce and reintroduce problem solving strategies in a variety of contexts and combinations.

Exploration and Discovery

After the student has at least a basic mastery of the programming language (when they have some control of the tool), they should have the freedom to explore and play with ideas in ways that seem interesting and worthwhile to them (Clements, 1986; Papert, 1980; Perkins & Martin, 1986; Weir, 1987). Logo was developed to be a natural and motivational environment to support student explorations. Hoyles (1985) found general agreement in the literature to support this notion. She states "many researchers have noted that programming in Logo seems quite naturally to provoke pupils to become involved in experimental activity while they try to understand a new idea, process or procedure" (p. 26). Krasnor and Mitterer (1984) cite several studies indicating that an approach to Logo based on exploration and play could lead to enhanced creativity and problem solving. They believe that:

> [when a] child is encouraged to try out new means of achieving goals and to combine old means to achieve new goals ... the result is a more flexible and adaptive mastery of logo that a more regimented approach. ... In addition, adaptive attitudes toward failure and a willingness to debug problem solutions should be facilitated. (p. 140)

The experimental activity that students engage in during their Logo explorations is seen as one way of building mental models of how the world works. It gives them the opportunity to view concepts from several standpoints and allows them to create concrete metaphors out of these formerly abstract concepts.

Papert believes the discoveries students make while "messing about with ideas" are the most meaningful and long lasting possible. The knowledge, ideas, and experience gained in this fashion are powerful because they come from a context that was designed by, and therefore more likely to be meaningful to the learner. These experiences have additional power because they are derived from direct manipulation of ideas and concepts, not just through language about them. In this sense, exploratory activity is an essential element of Piagetian or constructivist (*e.g.*, Lochhead, 1985) learning. Knowledge is something that the learner constructs out of his or her own individual activity while being responsible for the choice and direction of the activity.

Students shouldn't see themselves as just "doing computers" or "doing Logo" in isolation. Self directed projects allow students to apply their programming skills to a variety of subjects and interests. Exploratory activity should emphasizes the cross-curricular aspect of programming and therefore help to expedite the transfer of skills and concepts to other areas. In describing this type of approach, Franz and Papert (1988) state that:

> through programming languages such as Logo, computers allow our students, within certain limits, to perform tasks that are difficult or even impossible to achieve with other materials. We emphasize that it is possible to create activities that connect many different students' interests to various curricular areas, and to connect "separate" disciplines to each other. Our goal was, and continues to be, to create learning situations in which connections are allowed to develop freely and to move in any direction, albeit many or even most of them unpredicted. (p. 416)

A programming curriculum that fosters exploratory activity encourages students to actively use knowledge rather than passively receive it. They work in a context in which they experience situations where their knowledge does and doesn't apply. They learn how solve problems in the context in which they arise. In essence, the exploratory activity helps them learn how to learn.

Continuity

Programming teachers, especially on the elementary level, often seem to be unaware of or ignore the previous programming instruction their students may have had (Bork *et al.*, 1986). As with most other subjects, a programming/problem solving curriculum should provide for continuity from grade to grade. Students shouldn't have to begin anew each year. There needs to be coordination to ensure that programming instruction builds on and expands what the students already know.

At the same time, a curriculum based on the comprehension, integration, and generalization of higher order thinking skills doesn't lend itself to a rigid linear progression. The curriculum can introduce tasks that will gradually require the use of more and more skills to complete. But it can't just focus on the mastery of subskills out of context. Early on, the students will need to participate in and understand the global task of completing a programming project. This will require the teacher or materials to take responsibility for the local subskills that the students have not yet mastered.

Language/Discourse

Not until programming teachers become programmers themselves will they be able to talk to their students about the act and process of programming in any knowledgeable way. Papert (1985) states that "developing a discourse is at the heart of developing a culture, and a more textured and knowledgeable discourse about Logo contributes to the Logo culture, the computer culture, and the learning culture in the broadest sense" (p. 55).

Soloway (1986) also believes that developing a programming discourse is at the heart of the process. He sees a computer program as having two audiences. The first is the computer that turns the program into a mechanism that dictates how a problem can be solved. The second is the human reader who needs an explanation as to why the program solves a given problem. The process of creating mechanisms and explanations is seen as crucial because it transcends the domain of programming and is used in all problem solving situations. Soloway uses the example of giving instructions to a visitor on how to get from the airport to an office as a mechanism and providing an explanation of why that particular route is an effective choice as an explanation. He believes when students develop the ability to create explanations as to why their programs work or don't work, they will be better able to explain and act on their confusions in related areas.

This hypothesis is supported by much of the thinking skills literature. Vye and Bransford (1981) emphasize that all new thinking skills programs should strive to make implicit thought processes explicit. Heiman and Slomianko (1986) define critical thinking as engaging in an internal dialogue that employs the skills of "raising questions, breaking up a complex idea into components, drawing on your own prior knowledge, translating complicated ideas into examples, and generating hypothesis" (p. 7). They believe poor problem solvers have difficulty in carrying on this internal dialogue. They cite research by Bloom and Broder (1950) and Whimbey and Lochhead (1981) indicating that thinking skills can be improved by making this internal dialogue more explicit.

Specifically, they state students should be encouraged to think aloud when solving problems and to work in pairs. By externalizing their thinking "errors and skipped steps can be readily identified: in addition, working in pairs helps students edit unsystematic thinking in themselves and others" (p. 8).

Hoyles (1985) relates this back to Logo programming by describing her Logo Maths Project. She states that discussion among students is:

> crucial for both coming up with "things to try" and for developing a meta-language to describe the features of an activity. In learning to communicate with their partner(s), pupils also learn to articulate and elaborate their thoughts and strategies; to listen to and work upon their partners suggestions; to make and challenge conjectures; and to attempt to clarify incompatible explanations and problem solving strategies. It is also hypothesized that discussion helps students understand the formality and syntax requirements of a programming language, and helps them "see" the merits of alternative ways of representing a given situation. (p. 26)

It is clear that a major focus of programming instruction should be on teachers and students working together to develop ways of expressing the ideas and problems they encounter while programming.

Social Interaction

Since discussion and explanation are important behaviors in a programming environment, students need to be given the opportunity to work together in pairs and small groups. In addition to encouraging students to articulate their thought processes, Presseisen (1986) cites the importance of group projects in helping students appreciate multiple points of view. Students working together tend to develop responses consistent with evidence, instead of searching for an elusive "one right answer."

Krasnor and Mitterer (1984) cite three major benefits of group projects in Logo. First, the negotiation and confrontation that results are important factors in the development of the cognitive skills of decentration and perspective-taking. Second, students have the opportunity to learn by observing the nature and outcome of others' actions. They can view the work of their peers as benchmarks for their own progress. Third, by having the opportunity to teach others, the students are given the opportunity to consolidate and practice skills.

In reviewing the literature on the effects of computer environments on social-emotional development, Clements and Nastasi (1985) found that in Logo environments:

> besides promoting cooperative work, these interactions may also enhance the development of problem-solving skills, effectance motivation, and metacognitive abilities. In fact, child-child interactions during Logo programming may be as significant for cognitive development as are the child-computer interactions. (p.27)

Group work in Logo has usually been more a result of insufficient resources than deliberate intent. The benefits of social interaction shouldn't be used as a justification for not providing students with the hands-on time necessary to gain experience and develop skills. Whether working alone or in a group, engagement in the process is essential.

Motivation

For students to learn anything from programming, they have to program. Motivation is essential to get students involved in the task of programming (Bork *et al.*, 1986). Krasnor and Mitterer (1984) believe "it is in the motivational component, perhaps, that Logo has a special advantage over other methods of teaching problem solving" (p. 141). Lepper (1985) notes that Logo-like computer environments offer precision and interactivity as well as the potential for challenge and fantasy involvement. This is thought to result in intrinsically motivating activities. Papert (1980) and others (Malone, 1981; Bull & Tipps, 1984; Lepper, 1985; Linn, 1985; Dalbey, Tournaire, and Linn, 1985) stress the importance of intrinsically motivating computer learning environments. In these environments motivation stems from the students' desire to design, develop, and complete their own projects. Intrinsic motivation usually means that students view their projects more as play than work or a requirement. This means they are free to explore, adapt, change direction, and pursue unanticipated developments. This differs from extrinsic motivation in which the students' motivation stems from a desire to get a good grade or please someone else. Extrinsically motivated students are usually less prone to exploration and discovery and more likely to follow prescribed paths.

Assessment

Since one of the key aspects of a Logo culture is intrinsic motivation, the culture needs to be non-judgmental. When students know their projects will be evaluated the focus of their activity will shift to meet some outside criteria, and will in turn, become less personally meaningful. The absence of direct evaluation or grading does not mean that a Logo culture should not include some form of assessment (*e.g.*, Bull & Tipps, 1984; Watt, 1984; Weir, 1987). Teachers need to become aware of the activity and progress of their students in order to structure the environment to provide the appropriate help, resources, and freedom to suit the individual student. Schools also need ways to confirm that their goal of improving thinking skills is being met.

A variety of approaches to evaluating student progress in programming/problem solving environments need to be developed and applied. No single test covers all higher order thinking skills, and as Presseisen (1986) states "paper and pencil instruments may not be the best way to address the assessment of the varied modalities that thinking can encompass" (p.25). The type of assessment used should depend on the particular goals of the learning culture. If the goal is to impart factual knowledge (syntax and semantics), then some type of direct evaluation of that knowledge may be appropriate. To assess student progress on a deeper level, teachers need to consider all the information and behavior available to them. Hoyles (1985) states teachers should use all observable work by the student (*e.g.*, written plans, rough work, and actual programs) as a powerful means of illuminating both how the problem has been perceived and how it is to be solved. In addition, approaches such as noting the amount and type of help students request, their ability to articulate a plan or problem, how they work with and/or assist peers, or their ability to work independently, can be used to access student understanding and progress.

Conclusion

If a major function of schooling is to develop independent creative thinkers, then the promise of programming needs to be fully and seriously pursued. Logo in particular and programming in general is in danger of being relegated to the scrap heap of educational innovations that promised much but delivered little. Like a variety of other educational innovations, it has suffered from misguided disseminations, research, and implementations. Yet, there are strong indications that if the focus of programming instruction is on the learning culture, not just on isolated aspects of it, cognitive gains will be realized.

This paper outlines the aspects of a programming culture that may be necessary to bring about cognitive change. It seems obvious that most schools are not currently in a position to implement a programming culture similar to the one described here. To do so would stretch their resources and ability much too thin. To date, we have seen generally low level applications with high level expectations. Salomon and Perkins (1987) describe the problem this way:

> In the long run, carefully designed curricula, improved programs of teacher training, and similar means may make possible the routine implementation of programming instruction that has an impact on cognitive skills. At present, every such effort appears to be a separate saga full of false starts, unexpected problems, and, most often, unsatisfactory results. (p. 164)

The programming culture described in this paper is not a cookbook for change but it can serve as guidelines for designing research, creating curricula, and evaluating current practice. If nothing else, it may help schools realize they lack the resources, time, and commitment to properly implement programming as a vehicle for developing higher order thinking skills.

References

Becker, H. J. (1987). The importance of a methodology that maximizes falsifiability: Its applicability to research about Logo. *Educational Researcher*, *16*(5), 11-16.

Bloom, B. S., & Broder, L. (1950). *Problem solving processes of college students*. Chicago: University of Chicago Press.

Bork, A., Pomicter, N., Peck, M., & Velso, S. (1986). Toward coherence in learning to program. *AEDS Monitor*, *24*(3&4), 16-18.

Bull, G. and Tipps, S. (1983-84, December - January). Problem spaces in a project-oriented LOGO environment. *The Computing Teacher*, 54-57.

Clements, D. H. (1985). Research on Logo in education: Is the turtle slow but steady, or even in the race? *Computers in the Schools*, 2, 55-71.

Clemments, D. H. (1986). Effects of Logo and CAI environments on cognition and creativity. *Journal of Educational Psychology*, *78*, 309-318.

Clements, D. H. and Nastasi, B. K. (1985). Effects of computer environments on social-emotional development: Logo and computer-assisted instruction. *Computers in the Schools*, 2, 11-27.

Collins, A., Brown, J. S., & Newman, S. E. (1988). Cognitive Apprenticeship: Teaching the craft of reading, writing, and mathematics. In L. B. Resnick (Ed.), *Cognition and instruction: Issues and agendas*. Hillsdale, NJ: Lawerence Erlbaum Associates.

Dalbey, J., & Linn M. C. (1986). Cognitive consequences of programming: Augmentations to BASIC instruction. *Journal of Educational Computing Research*, *2*(1), 75-93.

Dalbey, J., Tourniaire, F., & Linn, M. C. (1986). Making programming instruction cognitively demanding: An intervention study. *Journal of Research in Science Teaching*, *23*, 427-436.

Franz, G., & Papert, S. (1988). Computer as Material. *Teachers College Record*. *89*, 409-417.

Heiman, M., & Slomianko, J. (1986). *Critical thinking skills*. Washington, D.C.: National Education Association.

Hoyles, C. (1985). Developing a context for Logo in school mathematics. *Logo 85 Theoretical Papers*, *1*, 23-42.

Krasnor, L. R., & Mitterer, J. O. (1984). Logo and the development of general problem-solving skills. *The Alberta Journal of Educational Research*, *30*, 133-144.

Lepper, M. (1985). Microcomputers in education: Motivational and Social Issues. *American Psychologist, 40*(1), 1-18.

Linn, M. C. (1985). The cognitive consequences of programming instruction in the classroom. *Educational Researcher, 14*(5), 14-16; 25-29.

Lochhead, J. (1985). New horizons in educational development. In E. W. Gordon (Ed.), *Review of research in education*, vol. 12. Washington, D.C.: American Educational Research Association.

Malone, T. W. (1981). Toward a theory of intrinsically motivating instruction. *Cognitive Science*, (4), 333-396.

Mandinach, E. B. (1986, April). *Aspects of programming courses that foster problem solving.* Paper presented at the annual meeting of the American Educational Research Association, San Francisco. (ERIC Document Reproduction Service No. ED 267 908)

Mawby, R. (1984, April). *Determining students' understanding of programming concepts.* Paper presented at the annual meeting of the American Educational Research Association, New Orleans, LA.

Michayluk, J. O. (1986). Logo: More than a decade later. *British Journal of Educational Technology, 17*, 35-41.

Papert, S. (1980). *Mindstorms: Children, computers and powerful ideas.* New York: Basic Books.

Papert, S. (1985). Computer criticism vs. technocentric thinking. *Logo 85 Theoretical Papers, 1*, 53-67.

Perkins, D. N., & Martin, F. (1986). Fragile knowledge and neglected strategies in novice programmers. In E. Soloway & S. Iyengar (Eds.), *Empirical studies of programmers* (pp. 213-229). Norwood, NJ: Ablex.

Polya, G. (1957). *How to solve it: A new aspect of mathematical method.* Princeton: Princeton University Press.

Presseisen, B. Z. (1986). *Thinking skills: Research and practice.* Washington, D.C.: National Education Association.

Psotka. J. (1985). *Reflections on computers and metacognition.* Alexandria, VA: Army Research Institute. (ERIC Document Reproduction Service No. ED 263 206)

Salomon, G. & Perkins D. N. (1987). Transfer of cognitive skills from programming: When and how? *Journal of Educational Computing Research, 3*(2), 149-169.

Soloway, E. (1985). *Why kids should learn to program* (Research Report No. 420). Yale University: Computer Science Department.

Soloway, E. (1986). Learning to program = Learning to construct mechanisms and explanations. *Communications of the ACM, 29*, 850-858.

Vye, N. & Bransford, J. D. (1981). Programs for teaching thinking. *Educational Leadership, 39*(1), 26-28.

Watt, D. (1984). Creating logo cultures. *Pre-Proceedings of the 1984 Logo Conference*, 25-29.

Weir, S. (1987). *Cultivating minds: A Logo casebook.* New York: Harper & Row.

Whimbey, A., & Lochhead, J. (1981). *Problem solving and comprehension.* Philadelphia: Franklin Institute Press.

Wright, D. A., Melmed, A., & Farris, E. (1984). *Instructional use of computers in public schools.* Washington, DC: National Center for Education Statistics.

Opening Windows to Technology

Faye Wilmore
Dodson Elementary School
Hermitage, Tennessee

Abstract

The project "Opening Windows to Technology" will chronicle the outreach program developed to share innovative teaching and learning experiences gained through four years of working in a technology rich environment, the Apple Classrooms of Tomorrow (ACOT). ACOT is a long-term formative research project that explores, develops, and demonstrates powerful uses of technology in teaching and learning. It is a partnership between Apple Computer, Inc. and the Nashville Metropolitan Public School System. As a part of the outreach project, 18 Nashville Metropolitan Public School teachers spent three days observing, teaching, and exploring software and hardware in the ACOT. Each teacher wrote a proposal to be implemented in their respective schools. The stories of the implementation of these projects and the role ACOT has and is playing in the growth of technology will be shared.

Multimedia in Teacher Education

Ronald J. Abate
Cleveland State University

Abstract

The combination of video and computer technologies present educators with new options for enhancing the quality of teacher preparation. The College of Education at Cleveland State University is pursuing these options in its core teaching curriculum. Instructional materials which combine computer controlled videodiscs with traditional forms of instruction are presently under development and test. This presentation provides an overview of three computer controlled multimedia applications. The applications include: (1) the use of videodiscs for classroom presentations, (2) stand-alone interactive simulations, and (3) a hypermedia database which provides a link between what is described in textbooks and what occurs in the K through 12 classroom. The rationale for developing and implementing the multimedia application is detailed. Finally, the lessons learned from the implementation effort are described, and directions for future development are identified.

The Florida-England Connection: A Telecommunications Project for Kids

M. D. Roblyer
Florida A&M University

Abstract

In many ways, the Florida-England Connection Project is the Florida version of other projects around the country that link up American students with those in other countries to provide unique and profitable educational opportunities. It may be substantially different from the others, however, in that it is both (a) a curriculum development project that emphasizes integrating telecommunications activities with existing curriculum and (b) a research project that evaluates the impact of these activities on student attitudes and achievement. The Panhandle Center of Excellence at Florida A&M University has been funded by Florida's Department of Education during 1988-90 to establish the feasibility and desirability of using the state's education network (the Florida Information Resource Network or FIRN) to connect secondary (Grades 7-12) schools in Florida with those in the United Kingdom. The purpose of those long-distance connections is to give students learning opportunities not available through other methods or media.

In the view of many educators, easy access to the "Global Classroom" will be an important part of the educational system of the future. However, since costs and difficulties of implementing telecommunications-based instruction are still fairly large in comparison to other methods, one of the primary goals of the Florida-England Connection Project is establishing the instructional capabilities and benefits provided by these methods and resources. Results of the Florida-England Project to date indicate that exchanges between students from different countries may have a measurable impact on bringing them closer together, and that the more "real" the exchanges are, the more impact they have. This impact can be seen to have many practical implications for the future of global relations and, combined with evidence that telecommunications activities are at least as effective as other kinds of instructional activities, could serve as an acceptable rationale to other schools and districts for investing the substantial time and other resources required for implementing a cross-country connection.

The Use of Hypertext Methodologies in Teaching Pascal

Linda H. Rosenberg
Mathematics/Computer Science Department
Goucher College
Towson, Md 21204
(301) 337-6281

Charles Nicholas
Computer Science Department
University of Maryland/Baltimore County
Catonsville, Md 21228
(301) 455-3000
nicholas@cs.umbc.edu

Abstract

Hypertext technology is coming into wider use as new ideas and application areas are being accepted. This paper discusses a new data schema for hypertext. The schema has a tree structure as its basis, with parent/child linkage. Additional links may be used as needed. Nodes are used as text receptacles. The idea of two different node types is introduced. The first type, called "concept nodes", contain the main ideas or structures. The second type of node is associated with a concept node. These secondary nodes, called "information nodes", contain the information on the concept.

This schema is used in a prototype, Hyper-Pascal, for the development of Pascal programs by beginning programmers. In this system, the hypertext structure is hidden from the user, thus avoiding many problems usually encountered with hypertext. In addition to serving as an environment for preparing and storing Pascal code, the system provides on-line help with Pascal syntax and structure usage, without encouraging the user to become overly dependent on such aids.

The Concept of Hypertext

Hypertext is a methodology for reading or writing a non-linear structure. Historically, documents have consisted of long sequences of words that have been divided into lines and pages for convenience. A hypertext document is a non-linear document with an intrinsic logical structure. These hypertext documents have an inherent flexibility, allowing a user to access the information in any order. This is a qualitative change in the way people conceptualize information resources; a shift in perspective, using a new medium. Hypertext mimics the brain's ability to store and retrieve information quickly and intuitively by referential links.

A hypertext system is comprised of nodes and links. Nodes are generic boxes for holding any form of information, including text, data, graphics, audio, video, animation, source code, or a mixture of any other form of data. Links create an associative path between nodes and define a node's relationship to other nodes within the network, thereby providing the nonlinear organization in the hypertext document. Hypertext may be used as a computer-supported medium for knowledge representation, in which inter-linked nodes of information are displayed through the use of links.[Conklin]

Advantages/Disadvantages of Hypertext Systems in Education

Cognitive principles of learning are thought to be applied when using a hypertext system.[Dede] These learning principles involve active structure networks, schema theory, web learning, and generative processing. An example of this is the ability of a user to follow a web of connections in tracing knowledge and information scattered in multiple sources. It is also believed that hypertext constructs are similar to internal human cognitive representation.[Dede] The data representation in hypertext enables both the capture of divergent mental models, and their convergence into a coherent structure, by linking nodes into a semantic network. Hypertext systems have the capability to represent environmental significance and goal-directed context in an associational manner. The architecture facilitates the use of many chunks of multi-attribute knowledge needed for different type of users. These advantages enable hypertext's use to support learning in many areas.[Dede]

Additional advantages of hypertext that support a learning environment involve conceptual exploration, retrieval, training, retention, collaboration, customization and revision. Hypertext systems are powerful methods for research thinking, knowledge transfer, and design deliberation. They allow thinking to be focused on the hard critical parts of a problem, and help detect incompleteness and inconsistency by making assumptions and definitions explicit.[Conklin]

Hypertext also has some problems. One problem is getting "lost in space": the problem of not knowing where you are in a network, or not knowing how to get to some other place you know (or think) exists—a disorientation problem. Another problem is information myopia. Even if a system's response time is instantaneous, there is definite distraction or cognitive loading. This "cognitive overhead" is encountered when a user must choose which link to follow, while not loosing the mental image of their location within the network.[Dede] This problem would occur for example, if as a user is waiting for the next node to appear on the screen, or considering which node they wanted to branch to, the user lost their thought pattern and their location within the system, thereby becoming disoriented. The non-linear representation of text also presents difficulties for the user of a hypertext system. Text is normally a carefully organized collection of ideas with a coherent sequence or pattern. Readers depend on the structural parts to help them recognize the type of text (ie. an outline or paragraph); these expected references are usually removed in a hypertext system, possibly causing disorientation.

The use of an additional structure in the hyperdocument helps to solve these problems. Many existing hypertext systems allow the construction of hypertext with very little structure. This lack of structure means that the user might not be able to decide what to do next simply by knowing where he or she is in the hypertext at that moment. This, in turn, results in the disorientation and information myopia mentioned above.

Problems teaching Pascal programming

Teaching programming to beginning students involves many concepts, such as logic design, determining the correct syntax, and variable usage. Learning and understanding the syntax and correct statement usage is a major problem for students. Syntax diagrams in a text are useful, but may be difficult to locate and understand. Frequently the diagram is comprised of other complex diagrams, which then must be located for complete comprehension of the syntax for the statement in question. This often leads the student to incorrect guessing, or using simpler statements with which they are more familiar.

Several methods for guiding students through the syntax of Pascal have been tried. Several commercial packages, such as Turbo Pascal and MacPascal, are available. Another attempt was made at the University of Maryland at College Park, using a system called "Support". Support is a syntax-oriented editor, as opposed to the more common text-oriented editors. The system has knowledge of the Pascal language, and prevents the user from entering incorrect text by prompting the student with the program and statement formats. The student then writes his or her program by filling in the blanks on a program. However, problems were encountered when students moved to a normal Pascal compiler. They were lost without the prompts they had become accustomed to, and were not familiar enough with Pascal syntax to enter their program without them.

Prototype Hyper-Pascal

As we mentioned above, the hypertext schema, by supporting multiple learning properties and principles, is ideal in an educational environment. To demonstrate this, we address the problem of teaching programming to beginning students. The Pascal language was chosen since it provides a very structured program organization that could be fit to the hypertext structure easily. Our prototype, Hyper-Pascal, uses a hypertext system to assist students in writing Pascal programs. Help is built into the system, available on request, and, like an appendix in a text book, shows the proper syntax or structure of a statement. The Help section also explains the proper use of a structure or statement, such as a function, procedure, or a case statement. If the student uses a variable not previously defined, a warning message appears on the screen. The student must eventually enter the variable in the declaration section of the program, but the system does not enter it for them.

The Hyper-Pascal system is being implemented using the VIRGIL hypertext system, which is itself being developed at the University of Maryland at Baltimore County. VIRGIL is designed to support the construction of customized hypertext systems, and in particular it includes features that let users access the hypertext information without knowing the structure of the hypertext itself.

Hypertext Data Schema

We introduce addtional structure to hyperdocuments by using schemas, which describe the location and nature of certain types of information. The data schema for Hyper-Pascal is based on a tree structure. The nodes are of two varieties, concept and information. The concept nodes contain the main ideas, concepts or structures used in organizing the hypertext. Attached to concept nodes are the information nodes where the data is stored. In Hyper-Pascal, this data may be help (syntactical or usage information) on the concept, or it may contain source code, entered by the student, pertaining to the concept. All nodes have associated keywords that can later be used by the student searching for a specific program section or variable.

All nodes are assigned an identification number by the system. This assists in program order, editing, and browsing. This identification number is internal to the system, hidden from the user, and is used to implement the actual hypertext structure. The identification number identifies the node's level in the structure, it's relationship to it's siblings (order), and the node's type (concept or information). Information nodes have the same identification number as the concept node they are attached to, but conclude with a letter. All information nodes used in the Help function conclude with the letter

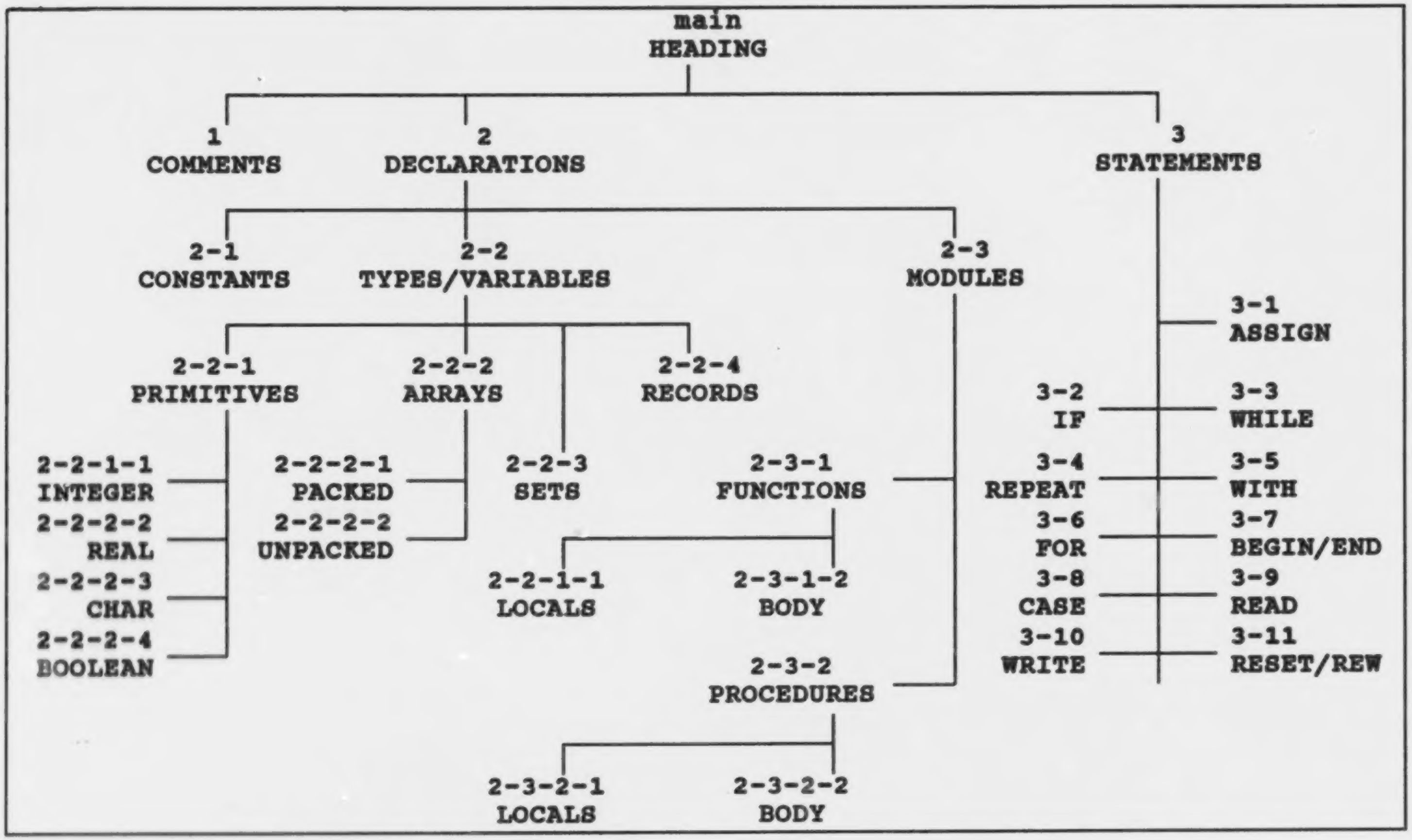

main
HEADING
1
COMMENTS
2
DECLARATIONS
3
STATEMENTS
2-1
CONSTANTS
2-2
TYPES/VARIABLES
2-3
MODULES
3-1
ASSIGN
2-2-1
PRIMITIVES
2-2-2
ARRAYS
2-2-4
RECORDS
3-2
IF
3-3
WHILE
2-2-1-1
INTEGER
2-2-2-1
PACKED
2-2-3
SETS
2-3-1
FUNCTIONS
3-4
REPEAT
3-5
WITH
2-2-2-2
REAL
2-2-2-2
UNPACKED
3-6
FOR
3-7
BEGIN/END
2-2-2-3
CHAR
2-2-1-1
LOCALS
2-3-1-2
BODY
3-8
CASE
3-9
READ
2-2-2-4
BOOLEAN
3-10
WRITE
3-11
RESET/REW
2-3-2
PROCEDURES
2-3-2-1
LOCALS
2-3-2-2
BODY

Figure 1. Structure of a Pascal Program in the Hypertext Schema

```
SAMPLE PROGRAM:

ln# Code

1    Program designct (orders, output);
2    (* comment *)
3
4    Const  Maxlength = 500;
5
6    Type   List = array[1 .. maxlength] of integer;
7
8    var    designlst : list;           (* comment *)
9           length : integer;           (* comment *)
10          count : integer;            (* comment *)
11          design : integer;           (* comment *)
12          orders : text;              (* comment *)
13
14   (**********************************************)
15
16   Function before (designlst:list;
17                    length, design: integer) boolean;
18   var    current :integer;      (* comment *)
19          found : boolean;       (* comment *)
20   begin (* comment *)
21          found := false;
22          current := 1;
23          while (current <= length) and not found do
24                 if design = designlst[current]
25                    then found := true
26                    else current := current +1;
27          before := found
28   end;
29
30   (**********************************************)
31
32   Procedure process;
33   (* comment *)
34   begin
35   while not eof(orders) do
36     begin
37     readln(orders,design);
38     if not before(designlst[length],length,design)
39                then
40                begin
41                    length := length + 1;
42                    designlst[length] := design
43                end;
44   end;
45   end;
46
47   (**********************************************)
48
49   begin (* main *)
50   length := 0;
51   reset(orders);
52   process;
53   writeln ('The following were ordered:');
54   for count := 1 to length do
55          writeln(designlst[count]);
56   end. (* main *)
```

Figure 2. Sample Program

A. Text nodes (program segments) conclude starting with the letter B and continue, using double letters if needed.

Links exist in two forms. The first is implicit through the identification number, and is used to connect parents and children in the tree structure. Explicit links connect separate portions of the program. These may be used, for example, to verify a variable's type, or that a procedure is declared before it is called.

Figure 1 shows the structure of a Pascal program as mapped into the new hypertext schema, including concept and information nodes and the use of identification numbers for the nodes. Program portions are contained in the hypertext schema as information nodes attached to the appropriate concept nodes. Exceptions are made for modules, such as procedures, functions and the main body of the program. These are retained as a unit in an appropriate information node. Statement syntax help is kept in the statement section of the schema, but separate program statements are not retained there. Figure 2 shows a short Pascal program that is stored in the Pascal Hypertext schema in Figure 3.

One additional item used in this schema is a listing of the program statements order. Since the program is not retained in the hypertext in the order in which it was entered, the system must be able to re-assemble the program in its original form. To prevent duplication of text, this listing contains only the line number and the node identification number where it has been stored. The only items the student ever sees on the screen are (1) their program (in the format in which they entered it), (2) warning messages, and (3) help when requested.

Advantages to Hypertext and Pascal

Applying hypertext to Pascal programming has many advantages for a beginning programmer, and none of the disadvantages normally found in a hypertext system. The student enters the program using hypertext in the same way as in any standard editor. If the student is unsure of a structure's use or syntax, they can request help on that concept. The help is immediately available, so there is no need to look it up in a text. Specific sections of code can be retrieved, viewing more than one section at a time if desired. Prompt error detection and on-line help support conceptual exploration, and encourage the use of unfamiliar principles. This is also positive feedback for the correct syntax. When a syntax error is made, it is immediately brought to the student's attention, and the student has the option of fixing it right away or later, after the statement or program segment has been completed. If the error is not corrected, it will again be listed when the program is compiled. This system of immediate error detection is superior to a batch-oriented processor that lists many different errors at one time, especially if one mistake has caused several of the others in the listing.

The common problems associated with hypertext systems, such as getting lost in the system, or disoriented and information myopia, do not occur in Hyper-Pascal. The student remains unaware of the hypertext structure, and the choices, movements, and changes made by the system are completely hidden.

```
hatch :: 0(program) --> 1(comments) 2(declarations) 3(modules)
parent 1
hatch :: 2(declarations) --> 2-1(constants) 2-2(type) 2-3(variable)
hatch :: 2-2(type) --> 2-2-3(array)
hatch :: 2-2-3(array) --> 2-2-3-2(non-packed)
hatch :: 2-3(variable) --> 2-3-1(integer) 2-3-5(text) 2-3-10(type defined)
parent 3
hatch :: 3(modules) --> 3-1(procedure) 3-2(function) 3-3(body)
hatch :: 3-1(procedure) --> 3-1-1(procedure 1)
hatch :: 3-2(function) --> 3-2-1(function 1)


cnode 0:: heading
inode 0A:: designct
inode 0B:: (orders,output)
cnode 1:: comments
inode 1B:: (* This program lists the designer numbers ordered *)
inode 1C::
      :: keyword :: blank line
inode 1D:: (********************************************************)
      :: keyword :: separator
cnode 2:: declarations
cnode 2-1:: constants
inode 2-1B:: maxlength = 500;
cnode 2-2:: type
cnode 2-2-3:: array
cnode 2-2-3-2:: non-packed
inode 2-2-3-2B:: list = array[1..maxlength] of integer;
cnode 2-3:: variable
cnode 2-3-1:: integer
inode 2-3-1B:: length: integer;       (* comment *)
        count: integer;          (* comment *)
        design: integer;         (* comment *)
cnode 2-3-5:: text
inode 2-3-5B:: orders: text;     (*comment *)
link ==> 2-3-5B == 0B
!# verify text variable listed in heading #!
cnode 2-3-10:: type defined
inode 2-3-10B:: designlst:list;  (* comment *)
link ==> 2-3-10B == 2-2-3-2B
!# verify pre-defined type #!
cnode 3:: modules
cnode 3-1:: procedure
cnode 3-1-1:: procedure 1
inode 3-1-1A::process
inode 3-1-1C:: (* comment *)
cnode 3-1-1-2 :: procedure body
inode 3-1-1-2A:: begin
           while not eof(orders) do
               begin
               readln(orders, design);
               if not before(designlst[length],length,design)
                   then
                   begin
                       length := length + 1;
                       designlst[length] := design;
                   end;
               end;
           end;
cnode 3-2:: function
cnode 3-2-1:: function #1
inode 3-2-1B:: Before
!# function name #!
inode 3-2-1C:: boolean
inode 3-2-1D:: (designlst:list;
        length,design:integer)
link ==> 3-2-1D == 2
!# verify declarations #!
cnode 3-2-1-1:: local variables
inode 3-2-1-1B:: current:integer;         (* comment *)
           found: boolean;           (* comment *)
cnode 3-2-1-2:: text
inode 3-2-1-2B:: begin  (*   *)
           found := false;
           current := 1;
           while (current <= length) and not found do
               if design = designlst[current]
                   then found := true
                   else current := current +1;
           before := found
       end;
cnode 3-3:: main body
inode 3-3B:: begin (*main*)
       length := 0;
       reset(orders);
       process;
       writeln ('The following were ordered:');
       for count := 1 to length do
           writeln(designlst[count]);
       end.
```

Figure 3. Application: Pascal Program

Conclusion

Using a hypertext data schema when writing Pascal programs provides several benefits to students. It enable them to have immediate on line help with syntax and statement usage. By storing the program in parts, such as variable, constants, function, procedures, etc, these can be retrieved separately for review or editing. Since there is no automatic prompting, this system does not handicap a user when working with a different Pascal editor, but encourages learning of the syntax.

Even though the hypertext structure is hidden from the user, many of the educational benefits of hypertext still apply. Cognitive principles of learning such as structure networks, schema theory and web learning, are being utilized. Hypertext's support of learning environments has been enhanced through the use of a schema. The ability to track a specific variable through the use of the search function allows a student to follow a web of connections through multiple program segments.

References

Beeman, William O., Anderson, Kenneth t., Bader, Gail, Larkin, James, McClard, Anne P., Mcquillan, Patrick, Shields, Mark, "Hypertext and Pluralism: From Lineal to Non-linear Thinking", *Hypertext '87 Papers*.

Charney, Davida, "Comprehending Non-linear Text: The Role of Discourse Cues and Reading Strategies", *Hypertext '87 Papers*, pp 109-120.

Conklin, Jeff, A SURVEY OF HYPERTEXT, Software Technology Program, *MCC Technical Report, Number STP-356-86, Rev.* 2, December 3, 1987.

Dede, Christopher, "Role of Hypertext in transforming information into knowledge", *NECC Conference Proceedings '88*, pp 95-101.

Fiderio, Janet, "A Grand Vision", *Byte*, October 1988, pp237-244.

Halasz, Frank G., "Reflections on NoteCards: Seven Issues for the Next Generation of Hypermedia Systems", *Communications of the ACM*, July 1988, Volume 31, Number 7, pp 836-852.

Nicholas, Charles, Lyle, James, "The VIRGIL System: A Framework for Hypertext Methodologies", Technical Report, Department of Computer Science, University of Maryland at Baltimore County, October 1989.

"Problems in Paradise", *Byte*, October 1988, 260-261.

Zelkowitz, Marvin V. "The Support Environment for the IBM PC", Department of Computer Science, University of Maryland College Park, Jan 1988, pp 2-13.

Teaching Programming Using the Karel the Robot Paradigm Realized with a Conventional Language

Roland H. Untch
Department of Computer Science
Clemson University
E-mail: untch@prism.clemson.edu

Abstract

An excellent method for introducing students to computer programming was described by Richard E. Pattis in his book *Karel the Robot*. By initially limiting the student's language repertoire to easily grasped imperative commands whose actions are visually displayed, the Karel approach quickly and effortlessly introduces the student to such concepts as procedures and the major control structures. However, some who have used the technique as a "quick-start" introduction to programming have noted some problems in the transition from using the Karel language to the conventional language used for the rest of the course (e.g. Pascal). By embedding the Karel programming paradigm in a conventional language, we have been able to eliminate these transition problems while retaining the pedagogical merits and spirit of Karel. This paper provides a brief review of the *Karel the Robot* programming paradigm, considers the transition problems, describes our novel use of Karel in a conventional language, and informally presents the results of using our version of Karel.

Introduction

Those of us charged with introducing programming to students do not have an easy task. The ability to analyze and solve problems is a skill not easily learned. To improve our student's problem solving and programming skills, we teach them a number of tools and techniques that enable them to "simplify" problems, that is, manage the complexity and size of the problems. We teach them about functional decomposition and stepwise refinement. We explain the concept of structured programming and discuss the merits of top-down development. In the process we introduce a number of tools, such as hierarchy charts and pseudo-code.

Unfortunately too much data and too many data types confuse the beginning student. Usually the scope of early programs is restricted in an effort to lessen the data complexity. The result, unfortunately, is programs that are often so simple as to be trivial. The early programs that we have traditionally been forced to assign (for instance programs to calculate mortgage payments, determine prime numbers, or compute Fibonacci series) do not require the use of "modern programming techniques". It is consequently difficult to establish the need and practice the use of the work management and design tools that the students will ultimately need.

Many have wrestled with this difficulty and a few satisfactory solutions have emerged [1, 2, 3, 7]. A rather novel solution was developed at Stanford University by Richard E. Pattis and is described in his book *Karel the Robot: A Gentle Introduction to the Art of Programming* [6,5]. What Pattis did was develop a robot programming paradigm, called **Karel** (pronounced "Carl"), that was entirely imperative. To quote from the preface of Pattis' book: "The careful omission of variables and data structures from Karel's language...allows the immediate exploration of the rich domain of abstraction and control structures." Having used this approach since early 1985, we have found that he was correct.

Others have reported on their use of Karel. Lt. Colonel Kenneth L. Krause states that at the United States Air Force Academy, "Karel proved to be enormously successful, the value of which far exceeded that of a mere motivator. Students easily grasped the subtleties of Karel and his language. They displayed impressive capabilities to employ top-down design/stepwise refinement techniques in solving relatively complex problems. They gained a solid appreciation of language structure, programming errors, and program behavior. In short, Karel lived up to all the claims of the author and represents a powerful pedagogical tool." [4]

Weaning students from Karel, however, causes some problems. It is a solution to those problems that we intend to present. In the following sections we will briefly review the *Karel the Robot* programming paradigm, describe the transition ("weaning") problems, present our method of using Karel in a conventional language, and indicate how this method addresses these transition problems.

What is Karel the Robot?

Karel is essentially a programmable cursor that can move across the flat world of a CRT screen. Shown on the screen is a gridwork of vertical and horizontal lines (avenues and streets) that form intersections or street corners. Karel is restricted to moving from street corner to street corner, one such move at a time. Additionally, Karel can pivot 90 degrees to the left when requested. Karel can only face North, South, East, or West and can always determine which direction he is facing.

To add variety to Karel's environment, Pattis added *beepers*, flashing symbols that Karel can detect ("hear"), pick up, carry, and put down. It is possible to program Karel to locate beepers, transport them, or place them in some graphic pattern. To bound Karel's environment there are *wall sections* that can be placed between streets and form impenetrable barriers. Karel can detect ("see") wall sections that are immediately to his front or sides. These wall sections can be used, to give two examples, to form obstacle courses that Karel must navigate or to represent hurdles that Karel must "jump" in a hurdle race.

Karel initially understands only five imperative commands: `move`, `turnleft`, `pickbeeper`, `putbeeper`, and `turnoff`. These are Karel's so-called primitive instructions. When these commands are executed in a Karel program, the results are depicted on the student's screen. A `move` instruction, for instance, will graphically show Karel moving from one street corner to the next. Should a wall section be in the way, however, Karel will signal the message "`Error Shutoff`" in protest and terminate execution of the program.

New instructions can be defined to extend Karel's vocabulary. For example, to define a `turnright` instruction one would write:

```
DEFINE-NEW-INSTRUCTION turnright AS
BEGIN
    turnleft;
    turnleft;
    turnleft
END
```

This definition must be repeated in each program that wishes to use a `turnright` instruction.

Karel is able to respond to the elements in his environment by testing a fixed set of predicates. The predicates Karel can evaluate or test are:

```
front_is_clear              front_is_blocked
left_is_clear               left_is_blocked
right_is_clear              right_is_blocked

next_to_a_beeper            not_next_to_a_beeper

facing_north                not_facing_north
facing_south                not_facing_south
facing_east                 not_facing_east
facing_west                 not_facing_west

any_beepers_in_beeper_bag
                            no_beepers_in_beeper_bag
```

(Note: Pattis used hyphens as separators within identifiers; we have changed these to the more commonly used underscore.)

These predicates are used in Pascal-like control statements. For example, when moving from corner to corner in a hurdle race, Karel could alter its actions based on whether a hurdle (wall section) is immediately in front of him or not. The code for this would resemble the following:

```
IF front_is_clear
    THEN move
    ELSE jump_hurdle
```

where `jump_hurdle` is presumably some instruction that the programmer has defined using the mechanism previously described.

Karel programs are either manually executed or run under a Karel simulator. The simulator is generally a simple but complete programming environment containing both an editor and an interpreter for Pattis' Karel language. Thus students using the simulator must first learn both this special pedagogical language and the commands of the simulator environment before they can test their logic.

The problems of transition

Once the basic features of Karel are mastered, and the student thoroughly acquainted with the programming techniques mentioned in the introduction above, the student is next taught to program in a conventional programming language, such as Pascal or Modula-2. Unfortunately this transition can be difficult. Some students become frustrated. Comments such as "bring back Karel" [4] are occasionally expressed. The transition is difficult because the student is simultaneously asked to learn a new language *and* a new operating environment *and* is presented with a new domain of problems.

Although Pascal-like, the Karel language is not Pascal. As such it is fraught with minor syntactic differences that must be identified and assimilated by those learning Pascal. This process can be maddening, especially to students who are still uncertain of their programming skills. And, if the language to be learned is not Pascal, the frustration level mounts. Students subsequently learning C or Ada, for example, are often confused by the use of the semicolon as a statement terminator instead of a separator. Finally, what skill the student acquired, in deciphering error messages produced by the Karel simulator, is of little avail with the new compiler.

This frustration with mid-course "retooling" is made worse by the need to learn a new editor and operating environment. We are all personally familiar with feelings of impatience and frustration when working with an alien editor or operating system. (Confess—how many of us run an old fashioned but familiar editor, like vi or spf or emacs, on our personal computers?) Such feelings can be especially disheartening to a beginning programmer. Moreover neither the student nor the instructor can afford to spend much time on delving into the details of this new environment. At this point in the course the instructor must use the established momentum to discuss other topics.

Similarly, students who may have felt free to experiment with an editor at the beginning of the term now have other demands on their time.

The greatest problem in the transition, however, comes from the underlying reason for the transition. Except for the implicit data object that is Karel's world, the two-dimensional screen, the student has not been taught how to declare and manipulate any data. Consequently the conventional language is introduced as a vehicle for teaching about variables, expression evaluation, assignment statements and the like. Alas, the initial programs assigned in the conventional language lack the intuitive feel of the Karel programs. For example, a student could easily ascertain that a Karel program was incorrect when Karel, say, tried running into a wall. A program to calculate mortgage payments, on the other hand, is less easily verified. In fact, all too often students resort to "democracy" to validate their results; that is, they compare answers and the majority output wins.

The solution to these transition problems is, of course, DON'T demand the students learn a different language, DON'T demand the students learn to use a different environment, and GRADUALLY switch to other problem domains. All this can be accomplished by embedding Karel in an existing conventional language. This is what we have done.

A method of using Karel in a conventional language

Instead of viewing the five Karel primitives as statements in a language, we elect to view the Karel primitives as invocable procedures that manipulate the screen data object. When thought of this way, it is a relatively straightforward matter to implement them as such. (If you wish, you may consider the screen as an abstract data type. The Karel primitives are operations on this data type.) These procedures are stored in a library where they can be referenced by (linked with) the student's program code.

We then simply use the constructs and syntax of the underlying conventional language to build our Karel programs. To extend Karel's vocabulary, we use procedures. For example, to define the `turnright` in Pascal, we would write:

```
(* Pivot KAREL 90 degrees to right *)
procedure turnright;
    begin
        turnleft;
        turnleft;
        turnleft
    end; (* turnright *)
```

Similarly, the Karel predicates can be viewed as parameterless Boolean functions that return screen state information. Rather than implement them as such, it is advantageous to implement them as global Boolean variables whose values are set by the primitive procedures as they are executed. Not only is this somewhat more efficient, but some compilers demand that functions with no parameters nonetheless be invoked with an empty argument list. Thus instead of simply writing `front_is_clear` we would have to write `front_is_clear()` which adds a useless, and potentially confusing, set of parentheses. Using these Boolean predicates in conventional control statements, we are able to write code like the following Pascal example:

```
procedure sparse_harvest_to_wall;
    begin
        if next_to_a_beeper then
            pickbeeper
        while front_is_clear do
            begin
                move;
                    if next_to_a_beeper then
                        pickbeeper
            end;
    end; (* sparse_harvest_to_wall *)
```

It is convenient to add a sixth Karel primitive, `turnon`, to initialize the screen. The `turnon` primitive reads from an external file the information necessary to initialize the encapsulated screen data structure, initializes the Karel predicates, causes the screen to be displayed on the terminal, and returns. After that, execution proceeds much as it would under a Karel simulator.

Any necessary declarations of external variables and procedures can be hidden from the student by placing them in a text file that is included by some standard compiler directive. This "include statement" is accepted by the students as a given. This inserted code is not visible on the student's source listing. An example of a complete Karel program, as prepared by a student, follows.

```
(* An expanded version of the Stair Cleaning *)
(* Task program from Chapter 3 of Pattis.    *)

PROGRAM stairs (INPUT,OUTPUT,SITUATION,REPORT);
%INCLUDE 'KAREL:KAREL(PASCAL)'

(* Pivot KAREL 90 degrees to right *)
procedure turnright;
begin
    turnleft;
    turnleft;
    turnleft
end; (* turnright *)

(* Climb on to next step *)
procedure climb_stair;
    begin
        turnleft;
        move;
        turnright;
        move
    end; (* climb_stair *)
```

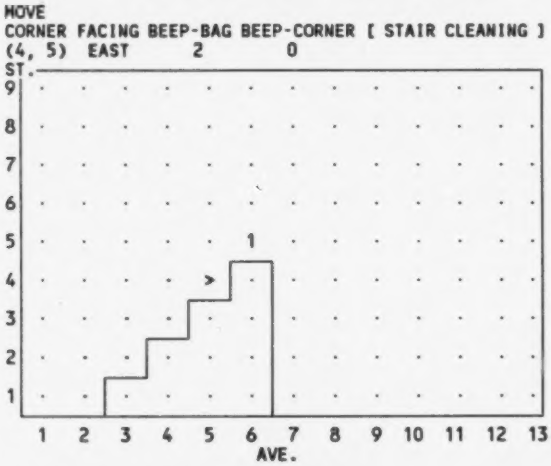
```
(* Attempt to remove a beeper *)
procedure pickbeeper_if_present;
    begin
        if next_to_a_beeper then
            pickbeeper
    end; (* pickbeeper_if_present *)

    begin (* main *)
        turnon;
        while front_is_blocked do
            begin
                climb_stair;
                pickbeeper_if_present
            end;
    turnoff
end. (* main *)
```

For the student's earliest assignments, the two lines

```
PROGRAM progid (INPUT,OUTPUT,SITUATION,REPORT);
%INCLUDE 'KAREL:KAREL(PASCAL)'
```

are given and used without explanation. The student is simply asked to change the program identifier name from assignment to assignment.

The following is a snapshot of the student's screen mid-way through the execution of the above program. (It has been edited slightly to fit on the page.)



```
MOVE
CORNER FACING BEEP-BAG BEEP-CORNER [ STAIR CLEANING ]
(4, 5)  EAST     2         0
```

Teaching with this version of Karel

Let us now examine how to introduce programming to students using this version of Karel. In addition to discussing course objectives and administration, the first lecture gives an overview of the computer system the students will be using. The students are assigned accounts, taught how to log on and off the system, and given a command that lets them execute a sample Karel program. Asking them to use the system prior to next lecture ensures some familiarity of the operating environment (lab locations, usage procedures, terminals, etc.) prior to the detailed presentation of that environment. Also, seeing Karel skitter across the screen rouses their curiosity. If, as occasionally happens, the accounts are not ready for the first lecture, a preliminary look at Karel is substituted instead. As the lectures continue with discussions of Karel programming, the labs can concentrate on teaching the students how to use the operating system, the hardware, and, especially, the editor. Exercises where the students enter and execute an existing Karel program are particularly helpful in building confidence and providing practice. Some of these programs are subsequently modified to illustrate new concepts. For instance, the use of procedures is introduced very early—at the end of the second lecture or the beginning of the third. To nail down this concept, the students are asked to define `turnright` and `turnaround` procedures. They then take a practice program and replace sequences of `turnleft` instructions with `turnright` and `turnaround` procedure calls as appropriate.

Since only a small subset of the language is being used at this point, the error messages produced are similarly constrained. Students rapidly learn to associate certain types of messages with certain mistakes. (On an indulgent day, we might say they "learn how to read the error messages".) As their language repertoire increases, they become increasingly adept at identifying the source of any lexical or syntactic errors.

By the fourth week of instruction the students have authored and run at least four or five complete programs (not to mention the exercise programs given to them). Not only does practice indeed make perfect, but this early amount of activity sets a pattern of work that continues throughout the rest of the term. Our students typically complete 12 to 14 programs in a semester, with the next to last program being a file handling program of approximately 1100 lines!

Even more importantly, the Karel programs by their very nature are well structured. Thus, when it comes time to formally discuss such issues as, say, stepwise refinement, the students already have an intuitive grasp of these issues.

Once procedures and flow control structures are well understood, it is time to discuss variables, expression evaluation, parameter passing, and the like. Since we are using a conventional language, we can immediately give illustrations of these concepts to our students. Moreover, graphic, Karel-style programming assignments using variables and numerical expressions are easily developed. For example, the following Pascal subroutine is part of an assignment where Karel needs to count the number of beepers on the current corner. The routine is invoked by `count_beepers_on_corner(number);`

```
procedure count_beepers_on_corner
                (var number : integer);
    var  i : integer;
    begin
        number := 0;
        while next_to_a_beeper do
            begin
                pickbeeper;
                number := number + 1
            end;
        for i := 1 to number do
            putbeeper
    end; (* count_beepers_on_corner *)
```

Not only are such assignments still easily understood by the student and the results readily apparent, but these problems retain the important element of being "big". By that, we mean they need a lot of subroutines. This is important! How else can we motivate the need for parameters and functions and top-down development?

Even long after the Karel section of the course has been completed, Karel examples come in handy. When discussing hierarchy charts, for example, it is easier to demonstrate with some relatively compact yet meaningful Karel-style modules rather than some contrived conventional ones. Certain types of exam questions are easier to write given the student's Karel background and the ability to thus make certain implicit assumptions about a problem.

All of the above illustrate the primary pedagogical advantage of using Karel in this way. *The students always add to their stock of knowledge and never need to relearn (or even unlearn) something.* Nothing is wasted (almost).

Summary

Karel is extremely useful in introducing students to computer programming. When implemented and taught as described above, certain pedagogical problems inherent in the simulator approach do not arise. Students do not experience as much frustration. Moreover, by eliminating certain extraneous issues, this approach is more efficient.

References

[1] Adams, J. Mack, Philippe J. Gabrini, and Barry L. Kurtz, *An Introduction to Computer Science with Modula-2*, D.C. Heath and Company: Lexington, Massachusetts, 1988.

[2] Drew, Mark S. and Shane D. Caplin, "Batch Logo—A Strategy for Introducing PL/I and Structured Programming to Gifted High School Students", *SIGCSE Bulletin*, vol. 16, no. 2, June 1984, pp. 13-16.

[3] Harvey, Brian, *Computer Science Logo Style*, The MIT Press: Cambridge, Massachusetts, 1985.

[4] Krause, Kenneth L., Robert E. Sampsell, and Samuel L. Grier, "Computer Science in the Air Force Academy Core Curriculum", *SIGCSE Bulletin*, vol. 14, no. 1, February 1982, pp. 144-146.

[5] Miller, Phillip L. and Lee W. Miller, *Programming by Design: A First Course in Structured Programming*, Wadsworth Publishing Company: Belmont, California, 1987.

[6] Pattis, Richard E., *Karel the Robot: A Gentle Introduction to the Art of Programming*, John Wiley and Sons: New York, 1981.

[7] Tomek, Ivan, *The first book of Josef: An Introduction to Computer Programming*, Prentice-Hall, Inc.: Englewood Cliffs, New Jersey, 1983.

Postscript

This paper describes our older VAX/VMS-based, Pascal version of Karel. Currently we are using a PC-based, Logitech Modula-2 version of Karel. The Modula-2 version is available through E-mail from the author. E-mail address: **untch@prism.clemson.edu**.

Turtle Graphics Can Enhance an Introductory Programming Course

Dean Sanders
Applied Computer Science
Illinois State University
Normal, IL 61761
Bitnet: nsander@ecncdc

Abstract

Discussions in an introductory programming course are normally accompanied by examples and assignments that generate textual or numeric output. The availability of graphics hardware makes it appropriate to incorporate computer graphics activities into several courses. The ease with which students learn the commands makes turtle graphics a reasonable basis for developing supplementary activities for an introductory programming course. Turtle graphics commands can be developed as extensions to the base language for the course. Activities based on turtle graphics can be particularly useful in a unit on procedures.

Introduction

From television to video games, today's students have grown up in a strongly visual environment. As a result, students are attracted to and interested in computer graphics. Now that computer graphics hardware is commonly available, it is reasonable to ask how graphics can be used to capture the students' interest and enhance their understanding of traditional computer science topics.

The success of the LOGO project at MIT (Papert, 1980) and some of its derivatives, notably Karel the Robot (Pattis, 1981), indicates that turtle graphics could be a reasonable way to introduce graphics activities into an introductory programming course. A useful set of commands is easily learned, and the anthropomorphic nature of the imaginary turtle makes it easy for students to develop and walk through solutions to problems.

There are two problems, one technical and one cultural, with using turtle graphics in a traditional computer science course. The technical problem, lack of software, is easily solved. As long as graphics hardware is available, a library of turtle graphics procedures can be developed to supplement any programming language.

The cultural problem can also be solved. There is a strong historical association between turtle graphics and LOGO, but except for a few experiments (Giangrande, 1988; Louden, 1989), LOGO has been associated with precollege courses. An unfortunate consequence of these associations is that turtle graphics is often viewed as elementary material that has no place in a traditional programming course. This view can be changed by providing turtle graphics procedures as extensions to the base language for a course, and by creating appropriate turtle graphics activities to enhance the content of the course.

The author has found that activities based on turtle graphics have considerable value in several courses. The remainder of this paper includes a summary the turtle graphics procedures that have been used by the author and describes how turtle graphics have been used in an introductory programming course.

Turtle Routines

Turtle graphics commands exist as extensions to some commonly used languages, but it is unusual to find an introductory programming course that uses one of these compilers. Fortunately, all is not lost. If the programming language contains a line drawing procedure, or at least a way to set a single pixel, then it is easy to create a useful set of turtle graphics procedures. The author has developed and used versions of the following turtle graphics procedures in FORTRAN, Pascal, and PL/I.

LeftTurn	(Degrees)	Turn left "Degrees" degrees
RightTurn	(Degrees)	Turn right "Degrees" degrees
Forward	(Distance)	Draw while moving "Distance" units
DrawTo	(X, Y)	Draw from current position to (X,Y)
MoveTo	(X, Y)	Move to (X,Y) without drawing
SetHeading	(Degrees)	Change heading to "Degrees" degrees
SetColor	(Color)	Change the drawing color
GetLocation	(X, Y)	Determine the current location
GetHeading	(Degrees)	Determine the current heading

Two important facts, the turtle's current position and heading, are maintained by the procedures. The heading is measured in degrees of counterclockwise rotation from the X-axis of a superimposed Cartesian coordinate system. The current position is represented in terms of a screen coordinate system. As presented to the students, the screen coordinates are real numbers that vary from (0,0) in the lower-left corner to (100, 100) in the upper-right corner. The actual screen coordinates are hidden within the procedures.

Using the Turtle

The first course in computer programming, be it in college or secondary school, normally includes quite a bit of work with procedures (subprograms). The list of topics for a unit on procedures normally includes abstraction, information hiding, initial and terminal states, parameters, and local variables. The following sequence of graphics case studies, presented as part of the classroom activities, provides a good introduction to the use of procedures.

The first case study begins with the basic concept of a procedure, progresses to the introduction of a local variable, and concludes with a demonstration of the importance of clearly specifying the initial and terminal states of a procedure. The second and third case studies introduce parameters and show how to create procedures that can be used to draw rectangles, circles, arbitrary polygonal shapes, and arbitrary curves.

Case Study 1:

> Write a procedure to draw a square that is 20 units wide and is centered on the screen.

Procedures SquareOne and SquareTwo shown below represent two ways to satisfy the requirements of this case study. As will be shown, there is merit to discussing, and generalizing, both procedures.

```
PROCEDURE SquareOne;
  BEGIN
    MoveTo ( 40, 40 );
    SetHeading ( 0 );
    Forward  ( 20 );
    LeftTurn ( 90 );
    Forward  ( 20 );
    LeftTurn ( 90 );
    Forward  ( 20 );
    LeftTurn ( 90 );
    Forward  ( 20 );
  END;
```

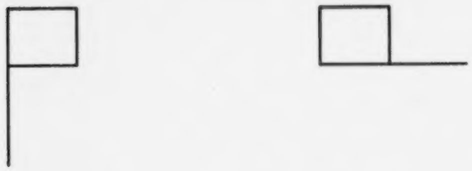
```
PROCEDURE SquareTwo;
VAR  K  : INTEGER;
  BEGIN
    MoveTo ( 40, 40 );
    SetHeading ( 0 );
    FOR  K := 1 TO 4  DO
        BEGIN
        Forward  ( 20 );
        LeftTurn ( 90 );
      END;
  END;
```

These procedures produce the same visual effect, but there are two significant differences. The most obvious difference is that SquareTwo uses a local variable. This opens the door to a preliminary discussion of the scope of a variable. A more subtle difference between these procedures is that they have different terminal states. This difference becomes evident when the following algorithms are used to draw a simple flag.

```
{---FlagOne---}       {---FlagTwo---}
SquareOne;            SquareTwo;
Forward ( 35 );       Forward ( 35 );
```

Although the flag drawing algorithms are apparently identical, an examination of Figure 1 shows that the results are quite different. The difference between the flags provides a good starting point for discussing the importance of clearly documenting both the terminal state of an algorithm and the assumptions about its initial state. The visual distinction between the two flags makes this point more emphatically than would numeric or textual output.

The graphical medium can be very intuitive. All students quickly realize that neither procedure for drawing a square is very useful yet many students have difficulty noticing a similar defect in procedures that lack graphical output. Discussion of how to make the procedures more useful leads to the concept of parameters. The two main lines of discussion are summarized in case studies 2 and 3 below.



Output from FlagOne　　Output from FlagTwo

Figure 1.

Case Study 2:

> Vary the size and location of the square.

A desire to vary the size leads to an introductory discussion of parameters. Procedure SquareThree (not shown) should be created by modifying SquareTwo to accept the length of a side as an input parameter. The new procedure now has both a local variable and an input parameter.

A desire to vary the location leads to a discussion of how one could specify the position of the square. The two most common ways are to provide the coordinates of the center or to specify the coordinates of one of the corners. Procedure SquareFour, shown below, accepts the coordinates of the center and the length of a side as input parameters.

It is worthwhile to continue this case study and develop a procedure that will draw a regular polygon of N sides. After a little guided discussion, it becomes clear that it might be easier to keep the entire image on the screen if the distance from the center to a vertex is given rather than the length of a side. Thus we have the following procedure which is very similar to

```
PROCEDURE SquareFour (Xc ,Yc, S : REAL );
VAR  A, B   : REAL;
VAR  K      : INTEGER;
  BEGIN
  A := Xc - S/2.0;
  B := Yc - S/2.0;
  MoveTo ( A, B );
  SetHeading ( 0 );
  FOR  K := 1 TO 4  DO  BEGIN
    Forward  (  S );
    LeftTurn ( 90 );
  END;
END;
```

SquareFour. An interesting experiment is to determine a relationship between N and R that can be used to draw a polygon that appears to be a circle of radius R.

Case Study 3:

Design a procedure that will draw a rectangle rather than a square and position the rectangle anywhere on the screen.

```
PROCEDURE Polygon ( Xc, Yc, R : REAL;  N : INTEGER );
CONST  Pi = 3.14159;
VAR  Theta, ExtAngle  :  REAL;
VAR  K                :  INTEGER;
  BEGIN
  Theta = Pi/N;
  A = Xc - R * SIN ( Theta );
  B = Yc - R * COS ( Theta );
  MoveTo ( A, B );
  SetHeading ( 0 );
  ExtAngle = 2.0 * Theta;
  S = 2 * R * SIN ( Theta );
  FOR  K := 1 To N  DO  BEGIN
    Forward  (   S );
    LeftTurn ( ExtAngle );
  END;
END;
```

This case study leads to the problem of specifying the location, size, and orientation of the rectangle. A little guided discussion should generate an agreement that the sides of the rectangle will be parallel to the edges of the screen and that the rectangle will be defined by specifying the coordinates of the lower-left and upper-right corners. This produces the following modification to SquareOne.

The power of a procedure, the ability to reference it with different values for the parameters becomes evident with an assignment such as one to draw a robot from rectangles. A sample robot is shown in Figure 2.

Additional Case Studies

After parameters have been introduced, the turtle graphics functions GetHeading and SetHeading can be used to introduce the important difference between value and reference parameters. These two procedures can also be studied to learn how one can hide something like the current heading from the user.

After arrays have been introduced, the turtle can reappear in a case study based on the problem of drawing a polygonal line given the coordinates of the vertices. This procedure, PolyLine, can be used in subsequent assignments that require the students to generate some sort of picture. As was the case with the polygon drawing procedure, PolyLine can be used to draw a smooth curve if N is sufficiently large.

Results

The author has used turtle graphics as the basis for developing case studies and assignments to enhance several courses including an introductory programming course. Because students have a keen interest in any form of computer graphics, these activities help revive enthusiasm in the middle of a somewhat difficult course. Students generally like the activities and many of them want to learn more about graphics. The visual impact of the graphics has helped several students understand concepts that they would have difficulty understanding with textual or numeric output. The author has observed that major concepts are learned and retained at least as well as they were in the absence of the turtle graphics activities. The simplicity of turtle graphics and the students' interest in any form of graphics, makes activities such as those described in this paper valuable additions to an introductory programming course.

```
PROCEDURE Rectangle( Xll, Yll, Xur, Yur : REAL );
VAR  Width, Height  :  REAL;
  BEGIN
 Width  := Xur - Xll;
 Height := Yur - Yll;
 MoveTo ( Xll, Yll );
 SetHeading ( 0 );
 Forward ( Width );
 LeftTurn ( 90 );
 Forward ( Height );
 LeftTurn ( 90 );
 Forward ( Width );
 LeftTurn ( 90 );
 Forward ( Height );
END;
```

References

1. Giangrande, E. and Allmaker, P. (1988). An Introductory Computer Science Course Using LOGO: A Case Study. *Proceedings of the National Educational Computer Conference 1988*. 304-307.
2. Louden, K. (1989). LOGO as a Prelude to LISP: Some Surprising Results. *SIGCSE Bulletin*. *21*(3). 35-38.
3. Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. NY:Basic Books.
4. Pattis, R. (1981). *Karel the Robot: A Gentle Introduction to the Art of Programming*. NY:John Wiley & Sons.

```
PROCEDURE PolyLine ( X, Y : REAL;  N : INTEGER );
  BEGIN
 MoveTo( X(1), Y(1) );
 FOR I = 2 TO N  DO
   DrawTo ( X(I), Y(I) );
END;
```

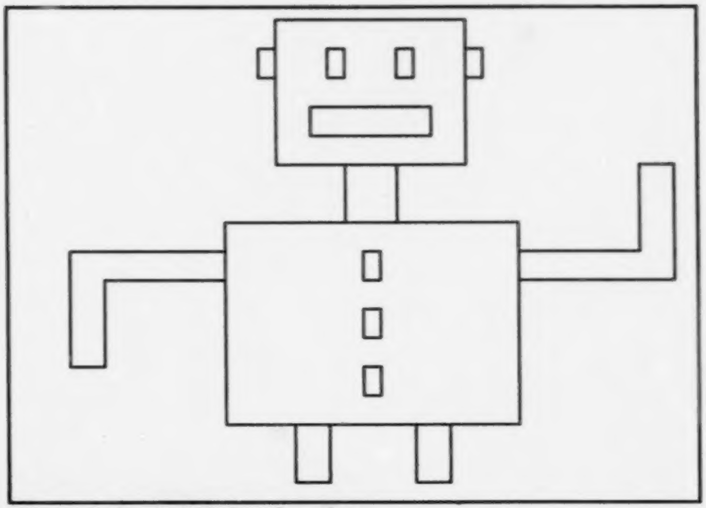


Figure 2. Typical Robot Creation

Kid Link: An Elementary Curriculum Telecommunications Project

Kendall Stall, Virginia Lawson, and Dolores Alexander
St. Charles Parish Schools
Luling, Louisiana

Abstract

Kid Link is a telecommunications project designed to apply and further discover the contributions that can be made to the elementary curriculum through the use of an information base. Telecommunications is integrated into the areas of language arts, science, and social studies. This project was designed and developed by a core group of educators, which included classroom teachers, instructional supervisors, principals, and a management information systems director from two school districts in the State of Louisiana.

This presentation will inform participants of pre- and post-survey results indicating areas of student growth and the effectiveness of the project in an elementary classroom setting, based upon the following objectives:

- Incorporate technology into the curriculum.
- Access information on current events and topics for research.
- Exchange, conduct, and compare results of science experiments.
- Learn of the diversity among peoples by sharing cultural information.
- Become active partners in the world community by thinking globally while acting locally.

The uniqueness of this program lies in the fact that technology has been effectively incorporated into the elementary curriculum. Working in a cooperative group setting within their respective classrooms, students conducted research, wrote reports, collected scientific data, and shared diverse social and cultural information.

LearningSphere 2008: A "Slice" of Tomorrow's School

Eileen H. Steele
Lafayette School Corporation

Nancy A. S. Miller
Indiana Department of Education

Abstract

The Indiana Consortium for Computer and High Technology Education sponsored two elementary level pilot programs that address some of the key issues in schools of the future: flexibility, choice, and learner-focused education that utilizes the available technologies to appropriately serve the needs of educators and learners. "Slice" began as a practical approach to the impracticality of financing a model of tomorrow's school in totality. A feasible approach would be to explore the same elements in the summer school program. A more relaxed environment with a modified time frame would provide room for a "Slice" of tomorrow's schools to be explored.

The LearningSphere 2008 concept features a learner-focused, flexibly scheduled, media and technology-rich environment for learning. Students and teachers have choices of topics, technologies, and times for learning and exploration in a multidisciplinary approach to education. As part of the total program several "high tech" and "high touch" experiences are provided for the learners: videoconferences, computers in learner's and learning guide's homes, telecommunications, CD-ROM, interactive video, robotics, electronic music, personalized newspapers, learning lunches, "spa," and day care.

The Regional Database: Critical Thinking in Social Science

Robert H. Summerville
Alabama Department of Education

Christine E. Drew
Lakeside/SSA, Pinson, Alabama

Abstract

In today's information society, facts and statistics bombard us without significance. Isolated facts hold no meaning. There is no real understanding of information; no knowledge is gained by reciting facts out of context.

Databases are information organizers that assist users in the manipulation of data. A database allows a user to develop hypotheses, search for information in support of a hypothesis, and draw conclusions based on facts. In short, databases allow people to take information and develop understandings that constitute knowledge. When people learn to use a database they have more control over their environment and more power to make good decisions.

Social studies teachers today are constantly frustrated by their inability to deliver the total content of the curriculum. Social studies is a relevant place to learn critical thinking skills. Through the use of databases, teachers of social studies can attack content and thinking skills while seeing students actively engaged in their own learning.

This past year, the Alabama Department of Agriculture, in conjunction with the State Department of Education, produced a database for delivering information about the demographics of Alabama. This database and packet of information can serve as a model for other states to provide regional information to their schools that will increase regional knowledge and computer literacy through social science classes.

The Satellite Office: An Integrated Telecommunications Project

Connie Stout
Texas Education Agency

Abstract

The Satellite Office was a natural outgrowth of the expansion of TEA-NET, the Texas Education Agency electronic network. The project involved teachers and students in successful and exciting activities for Microcomputer Applications classes. Microcomputer Applications is a course offered to Business Education and to Office Education students in high school.

A computer conference on GTE's Electric Pages provided a place for students and teachers to communicate and collaborate. The first messages transmitted to the electronic conference were letters students sent describing their school. Next the students used the telecommunications technology to share research papers. Later the students shared spreadsheet templates they developed.

TI-IN United Stars Network, a national satellite video-conferencing network based in San Antonio, Texas sponsored 2 hours of satellite time for two video conferences, one designed for teachers and the other designed for students in those classes. These video conferences gave teachers and students an opportunity to experience distance learning.

The teachers who participated in the project believed it was successful, and they were pleased with the quality of learning that took place in their classrooms. Although the project was designed primarily to illustrate how teachers from geographically separated areas can share curriculum materials, the teachers also discovered a supportive network of their teaching colleagues.

Connectivity—From the Macintosh to Other Computer Networks

Tara Gilmore Shlosman
Dr. Virginia Eaton
Northeast Louisiana University
Department of Computer Science
Monroe, Louisiana 71209-0575
(318) 342-3215
CNEATON@NLU.EDU

Abstract

During the summer of 1989 the Department of Computer Science of Northeast Louisiana University added a "Mac Lab," which houses eight Macintosh SE computers, to its other computer resources. With this addition, the Department became interested in learning which company has the networking software (and hardware) that would be best to use in linking the Macintoshes to the Ethernet network that already exists on the campus and to a future Personal Computer Local Area Network. By linking to the Ethernet network, the Macintoshes would have access to four VAX 11/750 minicomputers, a Micro VAX 3600 minicomputer, a Micro VAX II minicomputer, and an IBM mainframe. The Micro VAX II is currently being used as a UNIX machine. Access to the IBM mainframe is possible through a Series Network Applications gateway on the Ethernet. This paper presents the results of research into the best software and hardware to use for linking the Macintoshes to the other computers on campus.

Introduction

During the summer of 1989 the Department of Computer Science of Northeast Louisiana University (NLU) added a "Mac Lab," which houses eight Macintosh SE computers, to its other computer resources. With this addition, the Department became interested in learning which company has the networking software (and hardware) that would be best to use in linking the Macintoshes to the Ethernet network that already exists on the campus and to a future Personal Computer (PC) Local Area Network (LAN). By linking to the Ethernet network, the Macintoshes would have access to four VAX 11/750 minicomputers, a Micro VAX 3600 minicomputer, a Micro VAX II minicomputer, and an IBM mainframe. The Micro VAX II is currently being used as a UNIX machine. Access to the IBM mainframe is possible through a Series Network Applications (SNA) gateway on the Ethernet. This paper presents the results of research into the best software and hardware to use for linking the Macintoshes to the other computers on campus.

Apple Computer Inc.—AppleTalk

Apple Computer Inc. has a networking system available called AppleTalk. There are two cabling techniques that are widely used in an AppleTalk network: LocalTalk and EtherTalk (Sidhu, Andrews, & Oppenheimer, 1989, p. I-9). EtherTalk uses standard Ethernet technology, including thick or thin coaxial and twisted-pair cabling with data transmission at 10 million bits per seconds (M-bps). EtherTalk's fast transmission speed results in better performance. Also, EtherTalk can support more that 200 concurrently active AppleTalk devices (Sidhu et al., 1989, p. I-11).

LocalTalk, however, is built into the Macintosh's circuitry. Therefore, it is free with the Macintosh except for the cables. LocalTalk circuitry can range from twisted-pair to coaxial to fiber-optic. It is cost-effective for 10- to 15-node LANs and maintains good speed in that range (Abernathy & Rizzo, 1989). One must remember, however, that a node need not be one computer, but instead can be a printer, or any one of a number of other items.

Since an AppleTalk connector exists on the back of every Macintosh, and AppleTalk is part of the Macintosh operating system, AppleTalk is simple to install and learn. AppleTalk's cabling scheme uses the proprietary LocalTalk shielded twisted-pair technique operating at 230 Kilo (thousand) bits per second. LocalTalk supports 32 users over 1,000 feet, and bridges are available to double that length (Guterman, 1988). Yet AppleTalk can handle up to 256 stations (Scott, 1989). AppleTalk suffers shortcomings in speed and in working only with Macintoshes (Guterman, 1988). Since AppleTalk is built into every Macintosh used, Macintoshes are easy to connect to each other or to other computers of every size and make. This means that users need to add only a few inexpensive devices to the computers' motherboard and plug cables into the existing printer port of each machine (Young, 1988). AppleTalk offers extremely good communications for small work groups (Gralla, 1988).

AppleTalk is comprised of the cabling scheme called LocalTalk and Apple's networking software, AppleShare. The AppleTalk package of cables and software allows the integration of Macintoshes with IBM-compatible computers to use Macintosh IIs as file servers and to share Apple LaserWriter printers. The printers are considered stand-alone nodes on the Apple network. This AppleTalk connection between PCs and Macintoshes does not mean full integration, just file sharing and exchange. However, as more companies create both Macintosh and MicroSoft Disk Operating

System (MSDOS) programs, more files can be easily interchanged on AppleTalk. The easy-to-use AppleShare software is one of the best tools for simplifying PC-to-Macintosh communications, but AppleTalk offers extremely slow performance at high cost, since a Macintosh is needed as a dedicated file server (Derfler, 1988a).

In a discussion involving AppleShare, one must consider several facts, such as:

1. AppleShare is dedicated, meaning the Macintosh server cannot be used for other work and must always be accessed for file sharing.
2. AppleShare is based on the AppleTalk Filing Protocol (AFP).
3. AppleShare requires a system administrator.
4. AppleShare's access procedures are both complicated and poorly construed.
5. AppleShare proves somewhat faster in tests than Sun Microsystems Inc.'s TOPS.
6. AppleShare proves expensive, and print-spooling software and manuals are extra.
7. AppleShare does ensure future compatibility with AFP and user-interface consistency with other Macintosh-based LAN products (Bortman, 1989b).
8. The AppleShare network offers connectivity solutions to connecting Disk Operating System (DOS) machines to Apple's Macintoshes (Bortman, 1988).
9. AppleShare 2.0 allows five networked printers to be used at the same time and lets network administrators add, delete, or change access levels while the LAN is in operation (Forbes, 1988).

Apple Computer Inc.'s communications strategy is based on its proprietary AppleTalk communications architecture, which is similar to the International Standards Organization's Open Systems Interconnect (OSI) model. AppleTalk is a suite of networking protocols built into the Macintosh. Support of peer-to-peer communications between devices sets AppleTalk off from its PC LAN counterparts. The AppleTalk networks are relatively easy to set up because the network is self-configuring. But there are a number of ways to connect the Macintoshes to PCs because of the lack of a de facto communications architecture built into the PC. Macintoshes, however, can connect to Digital Equipment Corporation (DEC) and IBM hosts through terminal emulation software (Barron & Mitchell, 1988). Apple Computer supports connectivity through its support of OSI and Transmission Control Protocol-Internet Protocol. It is said that AppleTalk represents a viable interconnectivity technology that is supposed to receive support from the user community (White, 1988).

The AppleTalk layers correspond to the seven levels of the OSI model for communication systems, and products are available from other vendors that replace AppleTalk layers (Van Name & Catchings, 1988). The AppleTalk Filing Protocol (AFP) resides in the presentation layer of the OSI model, allowing users to share data files and applications from a shared resource such as an AFP file server. AFP-compliant file servers, such as AppleShare, are AFP servers that can be accessed by the AppleShare client software or by applications that make direct AFP calls to the operating system. AFP translates local file calls in a user's machine into network packets. The packets go over the network to the File Server Control Program (FSCP) on the file server machine, which can be a Macintosh or any computer configured as an AFP server. The FSCP then translates the AFP file calls into a form that the server machine's operating system can understand. Non-AFP compliance can cause problems that emphasize the need for standardizing the way applications talk to each other and to servers (Jones, 1989).

There are three basic ways to link Apple Macintosh and PC-based microcomputers on a common LAN: introduce PC-based microcomputers into a Macintosh LAN, integrate Macintosh microcomputers into a PC-based LAN, or provide gateways between Macintosh LANs and PC-based LANs. The first way was all that was possible until the introduction of the Macintosh SE and II, which provide slots for add-on cards (Van Name & Catchings, 1988).

A notable fact is that DEC and Apple have entered a cooperative agreement whereby they will integrate their network architectures by merging DECnet and AppleTalk protocols. The integration of the AppleTalk-DECnet (Ethernet) architecture is expected to be based on AppleTalk for Virtual Memory System (VMS). Users must presently decide between DECnet and AppleTalk protocols to connect Macintosh microcomputers and VAX minicomputers. Macintosh Ethernet connections are usually formed with equipment from Kinetics Inc. and Farallon Computing. Helix VMX is a VAX-VMS network implementation of the Macintosh-based Double Helix II from Odesta (Sustar, 1988).

Ethernet

Part of Apple Computer Inc.'s strategy for its Macintosh microcomputers is to integrate the Macintoshes with IBM PCs and DEC VAX minicomputers. Ethernet is one way to achieve this integration. Ethernet Macintosh connectors are higher-speed alternatives to Apple's LocalTalk network. There are two types of Macintosh Ethernet products; a gateway linking a LocalTalk Macintosh network to an Ethernet network and direct Ethernet connections for individual Macintoshes. Direct connection products work differently, depending on the type of Macintosh used. Steve Nelson, director of marketing for Kinetics Inc., says his company's EtherPort products line encompasses all the types of direct connections (Catchings & Van Name, 1989). In comparison with Apple's LocalTalk, Ethernet is both much faster at up to 10 M-bps and much more expensive than the 230,400 bits per second (bps) LocalTalk (Abernathy & Rizzo, 1989).

Novell—NetWare

The Novell NetWare 2.1 network operating system in many ways represents the state of the art. Currently, the versions available for connecting to other machines include three types of connections. The first is a product to support IBM's LU 6.2 peer-to-peer protocol (Musich, 1988). This product is a NetWare LAN gateway called NetWare LU 6.2. The gateway will connect NetWare to IBM host computers through peer-to-peer networking capabilities under IBM's SNA. The new gateway will be offered with the Token-Ring Gateway and the micro-to-mainframe program link called Token-Ring Multi as well as Novell's NetWare SNA Services product line (Keefe, 1988).

Another type is a connection to DEC VAX minicomputers. Novell Inc. introduced the NetWare VMS network operating system at the Dexpo East trade show in February 1988. NetWare VMS runs under the VAX-VMS operating system via an Ethernet card (Pompili, 1988). The NetWare VMS software for linking PC users on a NetWare network to DEC VAX minicomputers allows the VAX minicomputers to work as network file and print servers for IBM-compatible microcomputers running NetWare 2.0a (Brennan & Pappas, 1988). The program allows microcomputers on the LAN to share resources with the VAXes, and users can hot-key between terminal-emulation and their PC applications. Users are anticipating features which Novell is promising to provide, such as the ability to support Apple Macintosh microcomputers and third-party applications that exploit NetWare VMS's cooperative processing and version 5.0 of DEC's VMS operating system (Morrissey, 1988b). Novell Inc. intends to release a number of upgrades and enhancements for its NetWare VMS product. NetWare VMS runs on a VAX and allows PC-to-VAX communications on NetWare LANs with a variety of topologies. The upgrades should include the release of application program interfaces, the inclusion of Apple Macintosh support, the long-awaited Message Handling Service electronic mail feature, and full DEC routing support. The new releases are also expected to improve the product's initially lackluster performance (Morrissey, 1988a).

The third type of connection is a connection with the Macintoshes. Novell Inc. has announced NetWare 2.15 and NetWare for Macintosh, two new versions of its LAN operating system. The two programs combine to serve as Novell's entry into the Apple marketplace by giving users the ability to connect Macintoshes with most PC networks and operating environments currently supported by Novell. Macintosh users can communicate with Operating System 2 (OS-2)- and MSDOS-based machines and have access to all NetWare services. AppleTalk networks can exchange information with PCs running on Ethernet; Token-Ring, ARCnet, and other popular networks will be supported in future releases. The programs implement the AFP, allowing Macintosh users to view DOS and OS-2 files in the Macintosh's native mode (Scott & Forbes, 1988). NetWare for Macintosh adds AppleShare compatibility to one of the most popular DOS networks. To use the new product, NetWare Operating System 2.15, NetWare's most expensive option, is needed (Bortman, 1989a). The software allows Macintoshes on AppleTalk networks running AppleShare to share data and resources with PCs on NetWare LANs through connections to the network file server running Advanced NetWare 2.1. The system makes files on NetWare servers look as if they originated locally to Macintosh users, who are able to access information on either the NetWare server or any AppleShare server also connected to the network (Morrissey, 1988c). Even though NetWare 2.15 provides a link between a PC network and a Macintosh network, the link is restricted by the limitations of the AFP. The limitations include a restriction on the number of users that can access a single application simultaneously. The protocol limits multiple access to a limited number of files that are capable of 'multi-launch.' The NetWare implementation also does not have the utilities needed to execute some DOS files on the Macintosh side of the network. These limitations make the Apple-DOS connection useless for anything but print spooling (Gerber, 1989).

Sun Microsystems Inc.—TOPS

Sun Microsystems Inc. has a networking system available called TOPS. TOPS is usually compared with Apple Computer Inc.'s AppleShare. When a comparison between AppleShare and TOPS is made, the following differences are usually noted:

1. TOPS functions as a distributed system, meaning shared files are distributed.
2. TOPS uses Sun's TOPS Filing Protocol.
3. TOPS does not require a system administrator.
4. TOPS' less-extensive security system has fewer oversights.
5. TOPS outsells AppleShare for its easy use and installation, its low cost, and its equally adept co-functioning with PCs (Bortman, 1989b).

When TOPS is used with Apple computers (in particular, Apple Macintosh SEs), one should also note a few other facts, such as:

1. TOPS is a LAN system having excellent PC-Macintosh connectivity features that link PCs,

Macintosh computers, and peripherals. The system lacks the speed and features found on many PC-to-PC connectivity packages but is superb for 'PC-to-Macintosh-to-PC integration.

2. TOPS acts as a standardized environment, providing file access protocols that are universal among IBM PCs and compatibles, Apple Macintoshes, and UNIX-based machines.
3. TOPS-Macintosh is the TOPS software for the Macintoshes (Derfler, 1988b).
4. The TOPS AppleTalk LAN has proved very easy to install, and is fast and transparent. Indeed, TOPS' performance proved surprisingly spectacular (Shapiro, 1988).

It should be obvious from the above facts that TOPS networks offer a viable connectivity solution to connecting DOS machines to Apple's Macintoshes (Bortman, 1988).

TOPS (A Sun Microsystems company) has a variety of items currently available or planned for release in the near future for linking the Macintoshes to the IBM PCs and compatibles. One of these is TOPS-DOS 2.1. The TOPS-DOS 2.1 is part of a family of inter-operable network operating systems for UNIX, Apple, and MSDOS computers. The TOPS FlashTalk network interface card comes separately. The package's strength is its facility for allowing printer sharing and file exchange among computers with different architectures. A large drawback is its copy-protection scheme, which relies on serialization of work-station copies and requires each node to have a unique serial number in order to communicate. TOPS-DOS is best suited to those with multi-vendor networking needs (Derfler, 1989).

TOPS also offers TOPS 2.0, an operating system designed for flexibility and simplicity rather than speed. TOPS 2.0 offers a valuable way for companies to share data and printers among networks comprised of up to 32 PCs, Macintoshes and UNIX computers. This software-hardware set is best recommended for its PC-Macintosh file exchange, PC integration into Macintosh networks, and access for PCs to Apple LaserWriter products. The TOPS teleconnector is required for use of the network (Derfler, 1988c).

TOPS Netprint allows IBM PCs to use Apple LaserWriters directly via AppleTalk connections. For this, TOPS-FlashCard software is needed. The hardware link to AppleTalk is Farallon Computing's PhoneNet Connector (Derfler, 1988b).

Conclusion

Even though the information we gathered indicated that TOPS would probably be the best choice for connecting the Macintosh SEs to PCs, some of the experts with whom we discussed the subject indicated that there might be problems using TOPS to connect the SEs to the VAXes, while there should be no problems using AppleTalk and AppleShare to do so. Therefore, NLU has chosen to connect the SEs using AppleTalk and AppleShare. At a later date we will add a gateway to connect the SEs to the Ethernet network. We do not currently have a PC LAN. In the near future we plan to add a PC LAN using Novell Netware. We will probably link the PC LAN to the SE LAN through the VAX Ethernet LAN.

References

Abernathy, A., & Rizzo, J. (1989, May). Making a small net work. *MacUser*, pp. 151-158.

Barron, J. J., & Mitchell, R. L. (1988, August). The well-connected Mac. *Byte*, pp. S57-S68.

Bortman, H. (1988, September). Connecting the DOS. *MacUser*, pp. 108-123.

Bortman, H. (1989a, April). Netware for Macintosh ships. *MacUser*, p. 226.

Bortman, H. (1989b, May). Two for the node. *MacUser*, pp. 161-168.

Brennan, L., & Pappas, K. (1988, February 23). Novell links PCs on NetWare networks to DEC VAX minis. *PC Week*, p. 6.

Catchings, B., & Van Name, M. L. (1989, February 6). Ethernet Mac connections. *PC Week*, p. C14.

Derfler, F. J. Jr. (1988a, December 27). Building workgroup solutions: AppleTalk. *PC Magazine*, pp. 151-159.

Derfler, F. J. Jr. (1988b, May 31). Making connections: TOPS. *PC Magazine*, pp. 253-261.

Derfler, F. J. Jr. (1988c, June 14). TOPS. *PC Magazine*, pp. 176-177.

Derfler, F. J. Jr. (1989, March 28). TOPS-DOS. *PC Magazine*, pp. 127-128.

Forbes, J. (1988, March 1). Upgraded version of AppleShare LAN to offer CD ROM. *PC Week*, p. 6.

Gerber, B. (1989, January 30). PC-to-Mac networking has a long way to go. *PC Week*, p. 38.

Gralla, P. (1988, September 12). Consider needs before choosing PageMaker version. *PC Week*, p. 63.

Guterman, J. (1988, August 29). Networking Macs are becoming a more practical solution. *PC Week*, p. C21.

Jones, R. (1989, May). Following protocol. *MacUser*, pp. 114-119.

Keefe, P. (1988, February 1). Novell link opens SNA gate. *Computerworld*, p. 10.

Morrissey, J. (1988a, October 31). Enhancements await NetWare VMS. *PC Week*, p. 8.

Morrissey, J. (1988b, November 7). NetWare VMS: a beginning. *PC Week*, pp. 25-26.

Morrissey, J. (1988c, May 17). Novell-Mac software will debut in early June, sources say. *PC Week*, p. 5.

Musich, P. (1988, February 2). Novell NetWare makes use of LU 6.2 protocol. *PC Week*, p. C3.

Pompili, T. (1988, February 23). Novell NetWare users can now access DEC VAX servers. *PC Week*, p. C41.

Scott, K. (1989, June 5). Apple's connectivity blitz may meet longtime buyer needs. *PC Week*, pp. 1-2.

Scott, K., & Forbes, J. (1988, June 14). Novell raises NetWare bridge into land of Mac networking. *PC Week*, pp. 13-14.

Shapiro, E. (1988, January). Real-world answers: Reflex Plus, PhoneNet, and a TOPS network solve some practical dilemmas. *Byte*, pp. 205-207.

Sidhu, G. S., Andrews, R. F., & Oppenheimer, A. B. (1989). *Inside AppleTalk*. Reading, MA:Addison-Wesley Publishing Company, Inc.

Sustar, L. (1988, May 24). Apple and DEC join forces to integrate network architectures. *PC Week*, p. C19.

Van Name, M. L., & Catchings, B. (1988, May 17). PC-to-Mac LAN link varies with user needs. *PC Week*, pp. C29-C30.

White, T. (1988, June 13). Connectivity at Apple's core. *Computerworld*, pp. 49-50.

Young, J. S. (1988, January 12). AppleTalk options multiply as Mac gains corporate acceptance. *PC Week*, pp. C13-C17.

Incorporating Advanced Technologies into the Undergraduate Introductory Computing Course for Non-Majors

Daniel Farkas
Assistant Professor
1 Martine Ave.
White Plains, NY 10606

Abstract

This paper describes how advanced technologies (e.g. electronic mail, desktop publishing, hypertext, etc.) can be introduced in a meaningful way into undergraduate introductory computing courses for non-majors. The paper addresses the importance of learning advanced technologies, curriculum constraints which prevent their inclusion into existing courses, and techniques for incorporating them into the introductory course and across the computing (and non- computing) curricula.

Key Words and Phrases: Computer Literacy, Advanced Technology, Introduction to Computing.

Introduction

While there may still be some debate on what "introduction to computing" course curricula should be, the level it should be taught at and what audience it should be targeted to, most would agree (with room for rather wide degrees of variation) on what non-computing majors (e.g. business, liberal arts, etc.) should be able to do and understand at the end of a one semester introduction. With the recent development of powerful yet inexpensive desktop professional tools, it is desirable that students be more than simply exposed to emerging technology. Unfortunately, one semester is not really enough time to cover all the material necessary to develop computer literacy by any definition [fark89], [baro84], [dyck87], [pete87], [hala85], [bail87], and [spre85].

Furthermore, many of the skills currently taught to undergraduate non-majors for credit (word processing, spreadsheets, etc.) have questionable academic value. One view is that these topics should be covered in extra laboratory sessions and that class time be used for more traditional computing academic subjects (MIS, database technology, telecommunications, etc.).

The dilemma is how to insure that students are adequately prepared to meet the technology challenges of the future without turning the classroom into a trade school. One approach is to assign projects which require the student to use advanced technologies in the preparation of their course work. This is currently being done in the CIS101 Introduction to Computing class at Pace University with a great deal of success.

Advanced Technologies

In today's world, more and more routine office work is being replaced by the computer. When students begin their careers, they should be prepared to integrate with the office of the 1990's. This means that they should have exposure to desktop computer systems which incorporate the emerging applications of today. Students in the CIS101 class choose term projects which expose them to one or more of these new technologies.

The Personal Workstation

This involves the hardware and software functional characteristics of a desktop system, whether a PC, high resolution graphics workstation, or mainframe host terminal. Software features of interest include electronic mail, appointment and meeting scheduling, electronic calculator, etc.

Document Preparation

The CIS101 course at Pace covers word processing, but does not have the time to discuss some of the advanced features of document and report preparation. In the preparation of term papers, projects and reports, students will be able to use advanced features such as footnotes, different font sizes, document combination and the generation of tables of contents and indices.

Searching Online Databases

Given the wealth of information required in the professional workplace, this is an application that all students will have the need to use. They will have an opportunity to access a free University library service, PALS, which includes an online card catalog and an education database. Other information services include Dialog, BRS, Dow Jones Information Service, and CompuServ.

Presentations and Presentation Graphics

Another very important emerging area in computer applications is Presentation Graphics. This involves the creation of presentations with overhead transparencies, photographic slides, or computer based screen images. In all cases, the creation process is computer based and thus provides a wealth of features for producing dramatic visuals in manageable time. Each student will be required to participate in a class

268

presentation on their project using one of the presentation graphics packages available.

Desktop Publishing

Many organizations, large and small, create and produce their own newsletters, promotions, and forms. This emerging area of computer application provides for the integration of text and images in formats suitable for publication. Students who select a journal or newsletter project will have the opportunity to use and learn desktop publishing techniques.

Idea Processing

New software packages on the market are designed to help users organize their ideas from the outline through the rough draft and into a final product—usually a report or document. Idea processors allow for the introduction of ideas and marginal notes throughout the creation process. This type of software will be available for students to use.

Database Management for Research

In a world of increasing information storage and retrieval requirements, advanced usage (beyond the first semester literacy requirement) of user oriented database packages and Fourth Generation Languages (4GL) becomes increasingly important. Students will have access to some of the more popular software packages (DBASE III, RBASE 5000, FOCUS).

Data Communications

The need to understand data communications is becoming almost as important as understanding LOTUS 1-2-3 or Word Processing. In the near future, homes and businesses will be interconnected over private and public switched networks transmitting voice, data, and video information. For the CIS101 projects, data communications emphasis is on workstation (PC) connections and includes how computer communications work, communications hardware (e.g. modems), communications software (e.g. Crosstalk, SCOM), facsimile applications and the emerging area of interactive video/data processing (VIDEOTEX).

Hypertext

This topic incorporates the new ways in which text and images are interwoven to produce sophisticated teaching tools and human/machine interfaces. One of the term projects will allow students to explore this new technology.

The Introductory Course: CIS101

With the advent of low-cost personal computers which permit the creation of PC-classrooms, the undergraduate introduction to computers course for non-majors emphasizes a skills approach to literacy which, as indicated above, is under constant scrutiny, debate, and review. The course is divided into four major topic areas:

1. *Introduction to Computing*. This topic covers the basics of computer history, computer hardware organization, types and devices. In this part of the course the social impact of computers and topics in computer ethics are discussed.
2. *Introduction to problem solving for computing*. In topic two, the student learns how to develop algorithms, understand structured flowcharts, and develop a simple BASIC program. Topics from Systems Analysis, Systems Design and the Systems Development Life Cycle are introduced.
3. *Computer Software*. This topic provides an introduction to the IBM PC and the DOS operating system. It covers the following popular personal computer software applications: word processing (WordPerfect), spreadsheets (LOTUS 1-2-3) and database management (DBASE III Plus).
4. *Management Information Systems*. This topic covers a number of issues in organizational information systems:
    1. Mainframe, Mini and Micro Computers
    2. Transaction processing methods (batch, real-time)
    3. Point of sale
    4. Electronic Funds Transfer
    5. MIS organization structure and management
    6. Database Management Systems
    7. Networking and Data Communications
    8. Office Information Systems

Incorporating Advanced Technologies

After the introductory course with its built in time constraints, students have little opportunity to broaden their computing skills. They must either choose advanced courses in computing to broaden their literacy skills or depend on the use of the computer integrated into advanced courses in their disciplines.

Neither alternative targets a set of information processing and communication skills required in today's business world. By incorporating special projects into the curricula, students can learn many of these new skills not otherwise possible.

Students work singly or in groups of 3 to 4 students. Each student or group must chose a project from one of the options below:

1. Term Paper on one of the advanced technology topics described above. The paper is a 10 to 15 page survey of the technology area including what it is, how it works, and what is currently available in the marketplace.

2. Production of a journal in the group's academic field of interest (e.g. history, literature, etc.). Using advanced word processing and desktop publishing techniques, the group gathers contributions from other students and faculty to produce the journal.
3. Production of a Newsletter. Using available desktop publishing software and hardware, the group produces one issue of an "Advanced Technology" newsletter.
4. Production of the class Journal. All the term papers of the class are produced using a single word processor using a consistent set of formatting guidelines. This group collects the papers (electronically), and produces a journal which is distributed to the class at the end of the semester.
5. Hypertext course materials. The group chooses one of the technology areas and develops a Hypertext module for that subject.

Each group or student is required to give a ten minute presentation to the class on their term project. The presentation must incorporate overheads or slides using presentation graphics.

The advanced technologies are incorporated and reinforced in the following ways:

Workstations. All students use IBM PS/2 systems for their in-class instruction. The University electronic mail system is on an IBM 4381 accessed by 3270 terminals.

Electronic Mail. Students and groups communicate with each other and with the instructor using the University's electronic mail system.

Document preparation. All students use the same word processor and must use a comprehensive set of advanced features including margins, tabs, pagination, footnotes, index, table of contents, etc.

Online Database Research. Students are introduced to the online searching facilities available at the university library. They are encouraged to do bibliographic searches in the preparation of their project.

Presentation Graphics. Each group or individual student is required to make a presentation using one of the available presentation graphics packages.

Other Technologies. Students have the opportunity to explore any of the technologies through their choice of paper or project. Some exposure will be achieved during the in-class presentations. In many cases students can incorporate some of the tools during work on their project. For example, some students will use their own personal computers and a modem to communicate with the University electronic mail system, while others may choose a desktop publishing or hypertext project.

Status and Future Work

This approach has been successfully tried with an honors section of CIS101. These are non-computing majors with the same preparation as other students but part of a University program to attract particularly motivated students. All students actively participated in electronic mail communication and many did their research using the online database facilities in the library.

The only disappointment is that most students chose to work alone on term papers. It would have been interesting to try a number of the special projects. In subsequent classes, the students may not be given such leeway in chosing projects so as to encourage participation in the interesting special group projects.

Finally, there is no reason why the use of advanced technologies should be limited to the CIS101 class. Its role there is to reach students who will no longer be exposed to them. It is our hope that these technologies can be encouraged throughout the Information Systems curriculum, especially to student audiences who are already committed to trends emerging in desktop software and hardware.

Conclusion

There are two important goals of computer literacy (other than the pedagogical ones [pete87]).

1. The course should prepare computer literate students according to some objective organizational standard.
2. The course should be the beginning of a process of computer awareness and utility, not the end!

The computer is quickly becoming an important part of the graduate and undergraduate curriculum, and introductory computer literacy courses are helping students prepare themselves for advanced courses and future careers. True literacy generally occurs on one's own initiative after the first introductory course. It is hoped that by reinforcing emerging technology in student assignments, the process can be firmly rooted early in the students' career.

Bibliography

[bail87] Bailey, M.G., "Spreadsheets and Databases—Alternatives to Programming for Non-Computer Science Majors," *Sigsce Bulletin*, (19,1), Feb. 1987, pp. 499-503.

[baro84] Baron, N.S., "Should Every Learn Anything?: The question of Computer Literacy," *Sigsce Bulletin*, (16,1), Feb. 1984, pp. 108-114.

[dyck87] Dyck, V.A., J.P. Black, S.L. Fenton, "Beyond Traditional Computer Literacy,"

Sigsce Bulletin, (19,1), Feb. 1987, pp. 508-512.

[fark89] Farkas, D., "Computer Literacy and the Business Major," *Proc. Decisions Sciences Institute Eighteenth Annual Meeting*, Western Region Conference, Monterey Beach, CA, March 22-24, 1989, pp. 45-47.

[fuor89] Fuori, W.M., L. J. Aufiero, *Computers and Information Processing, 2nd Edition*, Prentice Hall, 1989.

[gust86] Gustavson, F.G., M.V. Sackson, Problem Solving and Basic, *SRA*, 1986.

[hala85] Halaris, A., L. Sloan, "Towards a Definition of Computing Literacy for the Liberal Arts Environment", *Sigsce Bulletin*, (17,1), March 1985, pp. 320-326.

[kroe87] Kroenke, D.M., K.A. Dolan, *Business Computer Systems*, Mitchell Publishing, Inc., 1987.

[pete87] Peterson, J.T., "Goals for and Lessons from a Computer Literacy Course," *Sigsce Bulletin*, (19,1), Feb. 1987, pp. 504-507.

[ryde84] Ryder, B.G., "A 'Hands-on' Approach to Computer Literacy," *Sigsce Bulletin*, (16,1), Feb. 1984, pp. 102-107.

[spen87] Spence, J.W., J.C. Windsor, *Using Micro Computers: Applications for Business*, Times Mirror/Mosby College Publishing, 1987.

[spre85] Spresser, D.M., "A moderate Approach to Computer Literacy," *Sigsce Bulletin*, (17,1), March 1985, pp. 327-331.

Chaplin: Combining HyperCard and Prolog Inference for Hypermedia-based Instruction

Víctor M. Mendoza-Grado
Purdue University
Current address: AT&T Bell Laboratories, 200 Park Plaza,
Room IHP 1F-513, Naperville, IL 60566
Internet: vmg@ihlpy.att.com UUCP: ...!att!ihlpy!vmg

Abstract

This paper describes issues concerning the augmenting of Apple's HyperCard system with the logic inference and capabilities of PROLOG and shows how this can help in developing more flexible interactive instructional systems. The combination helps HyperCard and Hypermedia to increase their logic and AI power while helping logic programming to make use of graphics and audio in a better user interface for hypermedia tutoring systems. Two cases are detailed: a natural language front-end for Hypermedia and an intelligent tutoring system that includes user modeling. The system runs on a Mac II under MultiFinder.

Keywords: Hypermedia, HyperText, HyperCard, Logic Programming, PROLOG, Intelligent Tutoring Systems.

Introduction

This paper describes a system that combines HyperCard [App87] and PROLOG and gives some case examples of its use in interactive educational systems. The weaknesses of HyperCard as an implementation of Hypertext have been documented extensively as well as the features that would be desirable in Hypermedia systems [Con87, Hal88]. Here we are concerned with the capabilities of HyperCard as a tool for developing Hypermedia-based tutoring systems that include logical inference capabilities. The idea here is to complement HyperTalk with theorem-proving capabilities to enhance the language and to make inferences about the run-time interaction with the user. Both systems benefit from the interaction; HyperTalk lacks Fifth-generation computing level constructs and most Tutoring Systems in PROLOG lack the sophisticated graphical database and hypermedia interface of HyperCard. Therefore, the combination of a PROLOG shell with the HyperCard paradigm can be a powerful one.

HyperCard and PROLOG

F. Halasz in [Hal88] argues in favor of the achieved gains in enhancing hypermedia systems with a rule-based engine. PROLOG is an AI language that supports logic inference and rule-based systems and has been used to develop expert systems. HyperCard is a hypermedia system that has several aspects in common with PROLOG:

a) *Prototyping*. For each of the arenas they serve, each tool has proven valuable for quickly developing applications.

b) *Simplicity*. HyperCard has been oriented towards the non-programmer and has simple syntax and semantics. PROLOG is based on pattern-matching and on a simple chaining rule of inference and control.

c) *Information management*. Hypertext nodes represent single concepts or ideas related by links to other nodes. A PROLOG program can be regarded as both a database and a program [Kow79, p. 125].

d) *Analogy of semantic networks*. Hypertext is a directed graph of textual elements organized essentially in the same way as semantic networks. Deliyani and Kowalski [DeK79] have already pointed out the natural relationship of logic and semantic networks.

e) *Educational applications*. This is one of the main orientations of HyperCard, although still lacking improvement. PROLOG design was not originally motivated by educational purposes, however, it has been successfully used as a teaching tool [NiD87, ScM86]. Interestingly enough, one of Piaget's [Pia53] assumptions is that propositional logic accurately captures the structure of mature human reasoning.

However, HyperCard does not have the inference capabilities of PROLOG. On the other hand, PROLOG by itself does not include graphics or sound capabilities (although a few implementations provide them).

Therefore, this project's goal is oriented to augmenting HyperCard with a PROLOG inference engine. The areas where HyperCard may benefit from this union are:

a) *User modeling*. User modeling is an important part of intelligent interactive systems [Fin87] and in particular of intelligent tutoring systems [Kas86], where logic programming has proven useful [KaM87]. This can also help to overcome one of the main drawbacks of hypertext systems, namely, lack of guidance to the user.

b) *Natural language capabilities*. The origins of PROLOG are very much rooted in natural language processing research and a pioneering natural language front end for database is Chat-80 [Per82] written in PROLOG. In a later section we explore further the possibilities for a natural language front-end for HyperCard derived from that system.

c) *Deductive database capabilities*. Knowledge representations in logic are called "deductive" because it is possible to obtain deductions from facts and rules when answering queries [GaM84, Llo83]. Deductive database theory includes much of relational database theory. Logic is used to express data, program, queries, views, and integrity constraints and there is usually a large amount of facts in proportion to the number of rules.

d) *Planning*. One of the problems in both Hypermedia and Intelligent Tutoring Systems is the lack of "global" knowledge about what is being taught or shown. Planning techniques are oriented to attack that problem [PeM86]. A clear and economical implementation in PROLOG of a planning system for dealing with interacting goals is Warplan [War74].

e) *Declarative programming*. Logic representation can be used to specify text format styles and conversions [Jam88] as declarations. It can also be used to specify picture items, such as button and field relationships [Per83], in describing the user interface. Other types of relationships and even control structures can be represented more easily in a declarative fashion.

f) *Additional programming paradigms*. PROLOG can easily support, through extensions or meta-programming, other programming paradigms such as constraint programming [JaL86], blackboards, frames and object-oriented modules [CoR89, Lee86, Zan84].

Interface Implementation

In this section a description of the implementation of the interface between PROLOG and HyperCard is made. First, we must point out that prior to system 7.0, the Macintosh has not supported a standard way to do interprocess communication (IPC) or, more appropriately, interapplication communication (IAC). In general, HyperCard can be extended through the use of XCMDs (commands) and XFCNs (functions) written in C or Pascal. This is a constraint as these are compiled languages, and most high-level work is implemented faster through prototyping supported by interpretative high-level languages such as PROLOG (which could be compiled too). This would create a blend that allows the developer to concentrate on the high-level ideas (as opposed to the more complex, Mac-dependent system-programming level issues). At the present time, there are several ways to implement such an interface:

a) Developing a PROLOG interpreter as a XCMD/XFCN.

b) Running PROLOG in the background (under MultiFinder) then use Appletalk to communicate with HyperTalk.

c) As in b) except that the communication is through the file system.

Therefore, it is required that the target PROLOG have a good access to the Macintosh Toolbox, either directly or, through external calls, in order to implement any of the above methods. We have selected a sound variation of method c) for our implementation. Chaplin consists of HyperCard running in the foreground (front window), while PROLOG runs in the background waiting for commands from the HyperCard stack. It then sends results back which are in turn read and processed by HyperCard. Inter-process communication is achieved through resource files. Messages are written as strings in a Macintosh resource file which (through Resource Manager calls) are later read by the other end. The specific PROLOG used for these experiments is LPA MacProlog, although the method is general enough to be used with any other similarly powerful PROLOG.

The control flow in the system can be depicted as in Figure 1.

Towards a natural language front-end for Hypermedia

The power of PROLOG for developing parsers has been discussed amply in the literature [CoH87] as variants of Context Free Grammars are supported naturally [PeW80]. The purpose of a parser in this kind of hybrid expert system shell would be to analyze the

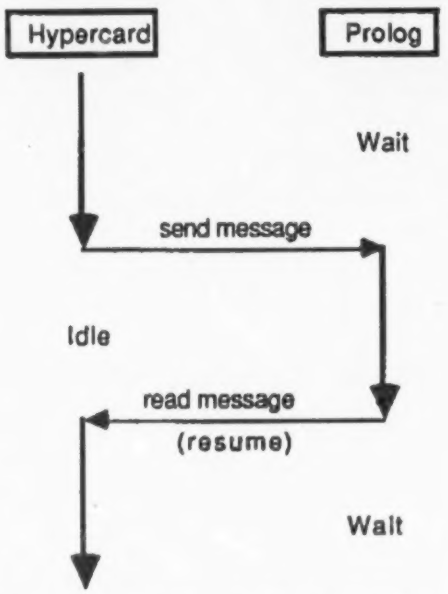


Figure 1. Control Flow in Chaplin Interface

text entered by the user (in a restricted domain) in the event that is not possible or desired to reduce it to a set of choices (where menus and lists would be more appropriate). Also, the front-end is used to query the database in a subset of English. Finally, this language handling capability is also being used to teach foreign languages, where PROLOG is in charge of checking the grammar, syntax and semantics of the language being tutored.

Chat-80 is a system that answers queries about geographical world data such as capitals of countries, oceans, rivers, and the types of responses are either lists of names, numeric or yes/no answers. For example:

What is the capital of upper_volta? (Ouagadougou)

How many countries does the Danube flow through? (6)

Is there some ocean that does not border any country? (yes)

For this particular implementation, the user is allowed to enter these types of queries into the message box provided by the standard HyperCard which the main script then translates to calls to the PROLOG interpreter running in the background. After the response is found and returned to HyperCard, the user has the option of displaying the graphical location of the response when possible (such as in the first type of query). In order to do this, the system keeps track of the locations of the geographical objects (countries, rivers, oceans, etc) described in the existing cards by means of the PROLOG database.

This system is being adapted to teach world geography and as a general tool for natural language input.

User modeling for interactive instruction

The second test case for this interface is for modeling the user weaknesses in intelligent tutoring systems. The modeling of the user is done by means of assertions about the answers and mistakes made during the course of the tutoring. Then, at certain points inferences are made, whenever possible, about possible paths of action to take to correct the weaknesses or failures found. The test case here involves a tutoring system for PROLOG beginners and includes a user modeling program similar to that of [Fin87].

Several studies (see for example [TaB87]) have been made about the problems encountered when teaching PROLOG programming to novices with no previous exposure to declarative languages. PROLOG includes many new concepts not found in strictly procedural languages such as pattern matching, backtracking, and the controversial cut operator. In order to introduce those concepts easily, the teaching strategy adopted is of utmost importance.

In this tutoring system, we begin by describing the fundamental aspects of the language, and continue on through the more advanced aspects of its usage, while providing drills of the material explained. The use of logical inference for teaching PROLOG is helpful to beginning programmers since its execution flow is a unique feature of the language. There are different models to convey the flow of control of logic programs. Among them: AND/OR trees, Arrow diagrams, and Byrd Boxes [PaB88], each with its own specific advantages and drawbacks.

Ideally, we want to be able to describe as succinctly as possible how control information is derived and how it is used in executing a program. For example, the AND/OR tree of the search space needs to be complemented with the database listing and the information for controlling search. However, that leads to confusing complexity and it is best replaced by a lesser amount of information, as needed. The other models mentioned provide a little more information than AND/OR trees (although with other drawbacks) and they are used according to the level of the programmer.

The tutoring system is able to select the appropriate model for each case presented, or program entered by the student, by maintaining a model of his/her characteristics during each run. The model maintained of the user relies on a 4-level classification of novice programmers. That level is upgraded as the user advances through the course and depending on the performance detected by making inferences about the responses entered by the user.

In this case, the use of HyperCard extends the computer mode of presentation by allowing animation of the execution models described above. In this way, the student can see how variables are given values, how choices are made and how execution of the case programs is performed.

Summary and Conclusions

The interfacing of HyperCard and AI languages such as PROLOG gives more power to the expressiveness of HyperTalk and to HyperCard as a development tool. It fills some of the holes of Hypertalk by providing more object-oriented capabilities to the stack by attaching declarative programs to each node (card). Also, it can add user guidance, natural language strengths as well as other AI capabilities to HyperCard and Hypermedia.

Chaplin permits the extension of the traditional text mode used by most tutoring systems. HyperCard includes some degree of the non-linear text presentation advocated by HyperText [Con87]. In addition, a more extensive video mode can be provided by making use of multimedia displaying more elaborate graphics and images as well as sounds.

Additional systems are currently being developed with this approach, most notably a version of the tutor described in [GoB87] to include videodisc and other media. A hypermedia-based grammar tutor for Spanish is also being developed using Chaplin.

Acknowledgements

I wish to acknowledge the help of Prof. Gordon L. Coppoc of Purdue University School of Veterinary Medicine in facilitating the necessary equipment while developing the ideas and software related to this project. Also, Prof. R. Lawler of the School of Education at Purdue deserves mention for his support.

References

[App87] Apple Computer Inc., *HyperCard User Guide*, 1987.

[CoH87] Cohen, Jacques and Timothy J. Hickey, "Parsing and Compiling Using PROLOG," *ACM Transactions on Programming Languages and Systems*, Vol. 9, No. 2, April 1987, pp. 125-163.

[Con87] J. Conklin, "HyperText: A Survey and Introduction," *IEEE Computer*, vol. 20, no. 9, Sep. 1987, pp. 17-41.

[CoR89] L. Console, G. Rossi, "Using PROLOG for Building Frog, a Hybrid Knowledge Representation System," *New Generation Computing*, 1989, pp. 361-388.

[CoB87] G. L. Coppoc, R. L. Bill, C. H. Lee, and V. M. Mendoza-Grado, "Development of a Frame-Based Intelligent Tutor for Pharmacokinetics," *Fifth Symposium on Computer Applications in Veterinary Medicine*, September 1987.

[DeK79] A. Deliyanni, R. Kowalski, "Logic and Semantic Networks," *CACM*, Vol. 22, No. 3, 1979, pp. 184-192.

[Fin87] Tim Finin, "GUMS A General User Modeling Shell," *User Models in Dialog Systems*, A. Kobsa, W. Wahlster, 1987, Springer Verlag.

[GaM84] H. Gallaire, J. Mikner, J. Nicolas, "Logic and Databases: A Deductive Approach," *ACM Computing Surveys*, 16, 1984, pp. 153-185.

[Hal88] F. Halasz, "Reflections on Notecards: Seven Issues for the Next Generation of HyperMedia Systems," *CACM*, July 1988, pp. 836-852.

[JaL86] J. Jaffar, and J-L Lassez, "Constraint Logic Programming," TR 74, CS Dept., Monash University, Australia, June 1986.

[Jam88] G. James, "AI and Automated Publishing Systems" *Text, ConText, and HyperText*, ed. by E. Barrett, MIT Press, 1988, pp. 15-24.

[Kas86] Robert Kass, "The Role of User Modeling in Intelligent Tutoring Systems," TR MS-CIS-86-58, Department of CIS, University of Pennsylvania, 1986.

[KaM87] K. Kawai, R. Mizoguchi, O. Kakusho, J. Toyoda, "A Framework for ICAI Systems Based on Inductive Inference and Logic Programming," in *New Generation Computing*, 5, 1987, pp. 115-129.

[Kow79] R. Kowalski, *Logic for Problem Solving*, Elsevier North Holland, 1979.

[Lee86] N. S. Lee, "Programming with P-Shell", *AI Expert*, Summer 1986, pp. 50-63.

[Llo83] J. W. Lloyd, "An Introduction to Deductive Database Systems," *The Australian Computer Journal*, Vol 15, No. 2, May 1983., pp. 52-57.

[NiD87] J. Nichol, J. Dean, and J. Briggs, "Computers and Cognition in the Classroom," in Rutkowska, J. and Crook, C. (eds.), *Computers, Cognition and Development*, John Wiley & Sons, 1987, pp. 135-152.

[PaB87] H. Pain, and A. Bundy, "What Stories Should we Tell Novice PROLOG Programmers", *AI Programming Environments*, R. Hawley (Ed.) Ellis Horwood, 1987, pp. 119-130.

[PeM86] Darwyn R. Peachey, and Gordon I. McCalla, "Using Planning Techniques in Intelligent Tutoring Systems," in *Int. J. Man-Machine Studies* (1986) 24, pp. 77-98.

[Per82] F. Pereira, *Logic for Natural Language Analysis*, Tech Note 275, AI Center, SRI International, 1983, also PhD Thesis, University of Edinburgh, 1982.

[Per83] F. Pereira, "Can Drawing Be Liberated from the von Neumann Style?," *Logic Programming and Its Applications*, M. Van Caneghem and D. H. Warren, Ablex, 1983, pp. 175-187.

[PeW80] F. C. Pereira and D. H. D. Warren, "Definite Clause Grammars for Language Analysis-A Survey of the Formalism and a Comparison with ATNs," *AI* 13 (1980) pp. 231-278.

[Pia53] J. Piaget, *Logic and Psychology*, Manchester University Press, 1953.

[ScM86] Z. Scherz, O. Maler, and E. Shapiro, "Learning with PROLOG—A New Approach," *Journal of Computers in Math and Science Teaching*, Fall 1986, pp. 31-37.

[TaB87] J. Taylor, B. Du Boulay, "Studying Novice Programmers: Why They May Find Learning PROLOG Hard," in Rutkowska, J. and Crook, C. (eds.), *Computers, Cognition and Development*, John Wiley & Sons, 1987, pp. 153-173.

[War74] D. H. D. Warren, "Warplan—A System for Generating Plans," DAI, Memo 76, Univ. of Edinburgh, June 1974.

[Zan84] C. Zaniolo, "Object-Oriented Programming in PROLOG," in *IEEE Symposium in Logic Programming*, 1984, pp. 265-270.

Using Videodisc Macrocontexts to Enhance Middle School Problem Solving Instruction

Michael Young, Nancy J. Vye, Susan Williams, James Van Haneghan,
Linda Barron, John D. Bransford, and Susan R. Goldman
Peabody College of Vanderbilt University

Abstract

This part of the session will focus on studies of our Learning Technology Center's efforts to anchor or situate problem solving instruction in complex, meaningful problem solving environments that are videodisc based. Our research group has reported at conferences initial baseline assessments of uninstructed students' abilities to formulate and solve our videodisc-based problems. Since students rarely have experiences such as these, it was not surprising to us that even high achieving sixth-grade students were very poor at this task. College freshmen were not much better.

In this presentation we will report on new videodisc-based programs and studies that have been conducted with regular education, chapter one, special education, and high functioning mathematics students. These studies evaluate model interventions and methods for integrating the instructional videodiscs into existing curricula. Student think-aloud protocols and pretest-posttest measures were used to assess the effectiveness of intervention strategies and compare them to control group interventions using more traditional instruction. Transfer of learning is the main focus of these studies and through the use of recent additions to the videodisc series, we have assessed student performance on near and far videodisc-based transfer problems. Initial results indicate positive transfer to multi-instructed problems requiring formulation and solution for groups instructed with experimental videodiscs, whereas control groups manifest relatively poor transfer.

Effects of Multimedia to Enhance Writing Ability

Charles Kinzer, Ted Hasselbring, Connie Schmidt, and Laurie Meltzer
Peabody College of Vanderbilt University

Abstract

This paper presents the results of work aimed at increasing the writing skills of sixth-grade, learning disabled students. Skills targeted are point of view, awareness of audience, cause and effect, and main idea with supporting detail. These skills are taught using a newspaper reporter format. This format was chosen because the sections of a newspaper fall conceptually into teaching the target skills and because students can create their own newspaper and thus write for a real purpose. In addition, students can write news accounts (based on video segments) that allow contextually grounded experiences to (1) be used as a basis for writing activities, and (2) be used as comparisons to actual news reports by experts.

Students are presented with real news footage (courtesy of NBC News) that has been pressed onto a videodisc. After seeing a news event, students are allowed to randomly access the video in order to determine answers to the what, where, when, why, and how questions that form the basis of a news story or to check facts or gather needed information. Lessons on point of view are taught using the various sections of a newspaper or news broadcast: sports, style, social, "hard" news, entertainment, and so on. News items are written on similar topics for various sections, as done in "real-world" newspapers, thereby necessitating shifts in perspective, audience awareness, writing style, and point of view.

A key aspect of this research is the provision of a videodisc-based context. This develops a contextualized approach to learning, and addresses a major issue in learning difficulties found in the target population: difficulties in learning that result from experiential deficits lack of background knowledge and/or inability to access shared contexts with a teacher.

Measures of effectiveness include pre- and posttests in the areas of writing with and without a video, in video comprehension, in comprehension of an orally presented topic, and identification of appropriate who, what, where, when, how, and why information. Data analysis indicates that students using a video-based, contextualized anchor as a part of writing instruction were better able to abstract main ideas, wrote more focused and appropriate products, and noticed significantly more information following the instructional program. Implications of using an anchored instructional approach with learning disabled students will also be discussed.

Effects of Videodisc Macrocontexts on Comprehension and Composition of Causally-Cohesive Stories

Victoria Risko, Charles Kinzer, Nancy Vye, and Deborah Rowe
Peabody College of Vanderbilt University

Abstract

This presentation describes a set of experiments conducted to examine the effects of video-based macrocontexts on students' ability to comprehend complex causal relationships among story elements and to use this information to construct well-formed stories. The studies described in this paper are part of a larger project funded by OERI/RIE's Literacy Research Award and extends our previous findings that macrocontexts—contexts that are semantically rich and can be used to invite students to examine concepts from multiple perspectives—enhanced students acquisition and use of new information and students' ability to generate story elements within well-formed stories.

A two-group repeated measures design was used across our experiments conducted during a 2.5-year period. Fifth grade students were assigned to either an experimental or control group using stratified random assignment according to performance level on the Stanford Achievement Test. Instruction for the experimental group was organized around a macrocontext provided by the film, *Young Sherlock Holmes*. This film provides an excellent model of problem solving; through the study of Sherlock, and invites students to generate questions for finding and learning story information such as setting (London during the Victorian era), character traits and motives, and events leading to goal resolution. A second film, *Oliver Twist*, and relevant texts were examined to build comparative and contrasting sets of information related to the setting of Victorian England, characterization and motives, and plot development. The control group was taught the same basic content, but the instructional procedures followed the traditional procedures suggested in the teacher guides for the school's texts.

The findings reveal large differences between the experimental and controls groups on several measures. Experimental students, across ability levels, were better able to identify causally related events, to describe character feelings and corresponding motives, and they wrote stories continuing plots that linked character actions and events to goal statements and goal resolutions. On several measures, data analysis revealed that the macrocontext instruction was especially effective for the low achieving students.

Using Videodisc-Based Instruction to Develop Computational and Conceptual Understanding in Elementary School Mathematics

Robert D. Sherwood, Ted S. Hasselbring, E. Jean March, and Jill C. Mertz
Peabody College of Vanderbilt University

Abstract

These participants will describe a 2-year funded research study of the implementation and effectiveness of using level one videodisc-based instructional materials to replace standard instruction in the topics of fractions, decimals, and percents in elementary and middle school mathematics. Two schools within a school system were designated experimental and control schools with the experimental school using the "Mastering Fractions" and "Mastering Decimals and Percents" videodiscs to replace standard instruction in these areas. Grade levels involved ranged from fifth to eighth.

An analysis of covariance design was used with scores from tests of computational ability provided with the videodiscs being used as pretest, posttest, delayed post, and retention tests. In addition, applications test involving fractions and decimals/percents were developed by project staff. Teacher and student interviews were also conducted to assess participants reactions to using this type of instructional media. Analysis yielded two major results. One, students in all grade levels in the experimental group were significantly higher in their scores on the tests of computational ability for post, delayed, and retention tests compared to the control group school. However, the test of applications of knowledge that emphasized the use of mathematical knowledge to solve word problems did not yield differences between the two groups. The results are consistent with the theme of the symposia, namely, the need to develop computational and conceptual understanding in mathematics by linking them in instruction.

A Writing Process for the 21st Century: Ideas for Using Desktop Publishing in the Elementary and Middle School

Sherah B. Nelson
Georgia State University
Gwinnett County Public Schools

Brevard Williams, III
Emory University
Gwinnett County Public Schools

Abstract

Participants will be introduced to desktop publishing within the context of an educational philosophy that incorporates the writing process, earlier envisioned by Graves and Hartse, with the enhanced capabilities of the computer. Since numerous recent research studies have indicated how the use of word processing and desktop publishing can affect the amount students write, the amount they revise, and student motivation for writing, this presentation will demonstrate some ways in which these ends can be achieved. Co-authoring will also be emphasized as an extension of the Vygotsian perspective that stresses peer collaboration.

The presenters will share interdisciplinary project ideas and examples of student work illustrating a "21st Century" expansion of the writing process based upon a whole language approach. Attendees will see how the introduction of high-interest, attention-getting graphics and font styles can bring new life to the student author's work in the elementary and middle school setting.

Music and Computers for Children with Special Needs

Nancy A. Norman
National Lekotek Center

Abstract

Computers have proven to be essential tools in providing children with special needs the opportunity to be independent, productive, and creative. The National Lekotek Center, through its COMPUPLAY program, has played a leading role in providing access to adapted technologies especially to young children.

A music and computer project, designed under the auspices of a volunteer Music Advisory Board, is currently being implemented into COMPUPLAY. This project provides children with special needs new opportunities to independently access music experiences through adaptive equipment and computers.

The objectives of this project include:

1) To develop educational play methods which enable children to experiment with music via the Apple IIGS, Unicorn board and/or MIDI device;

2) To develop and catalog play techniques for use by COMPUPLAY instructors, parents, and teachers;

3) To fine tune a music curriculum for the project;

4) To disseminate information on the use of adaptive music technology to special education teachers and therapists through training courses, workshops, and project presentations; and

5) To explore the use of the Apple Macintosh as a music tool for children with special needs.

An American Experience

Albert Ortiz
Desert View Elementary School
Sunland Park, New Mexico

Abstract

A group of bilingual students in this project are in the forefront of radio digital communication technology. Using traditional modes of radio communications, students have already benefitted in geographical knowledge and world affairs as well as by making friends in far off distance lands and in space.

This year the students are using computerized telecommunication in alliance with amateur radio in the classroom to become active in "Packet," a new digital communication strategy. Packet simply means sending information from the computer to a terminal node controller (TNC) connected to a radio transceiver, much the same way data is sent via a telephone. Data being sent by the computer is broken up into small pieces called "packets," hence from where the name comes.

Packet station at Desert View has several dramatic advantages over land line communications: (a) Packet itself is error free where other means of telecommunications are hits or misses caused by electrical interference or propagation; (b) Packet stations can send messages over long distances without any payment as the airways are free; and (c) there is a significant number of schools across the U.S. that are packet active.

Utilizing nontraditional methods of communications, students are recognizing new digital communication strategies: (1) Computer-Based Message System. Amateur Radio operators since mid 1970 have created a network for both bulletin board operation and computer-based message systems; (2) Keyboard conversation using radio link instead of a telephone; (3) Computer-base Baudot for transmitting text in place of ASCII; (4) AMOTOR to communicate in HF radio for telex; and (5) Receive radio facsimile pictures from weather satellites.

Amateur radio has contributed to modern digital communication based on Morse's telegraph system. The students in the classroom are also transmitting signals through a series of satellites known as OSCAR (Orbiting Satellite Carrying Amateur Radio), communicating with the Soviet Space Station MIR, and participating in SAREX, NASA's Shuttle Amateur Radio Experiment. Amateur radio in itself will allow the students to experiment and gain knowledge to improve and create new technological tools.

High Tech at Tech High

Joyce Perkins and Jim Hartman
El Paso Independent School District
El Paso, Texas

Abstract

"High Tech at Tech High" is the recipient of an Apple "Equal Time" grant for 1989. The award of $97,000 worth of hardware and software was given to the proposal focusing on Multimedia/Presentation Business Applications for Vocational Students. The presentation will focus on the writing of the proposal and the progress that is being made during this past school year. This project is taking place at Technical Center, a vocational technical school serving student populations from the El Paso Independent School District and outlying districts in the El Paso community. The students involved in the project are 11th- and 12th-graders enrolled in Business Computer Applications.

The presentation will include teaching materials and strategies that are being used that have enhanced the curriculum, student production work, student and teacher reactions to the project, along with the overall work encountered in working with Apple Computers, Inc. and others in the educational community. There will be many rewards that can be shared with educators across the nation. As a grantee we will share the little things that can help others in the acquisition of such grants and what is expected once the grant is acquired.

Technology Integration in Rural Settings: A Bridge Between Computer Applications and Local Curriculum

Jim Parry
TIE (Technology In Education)
Rapid City, South Dakota

Abstract

Access to and support from SD's technology resource center, referred to as TIE (Technology In Education), has allowed teachers in isolated settings to grow into computer users and local leaders. Decisions in these rural districts have become broad-based, logical choices leading to relevant computer applications that respond to local curriculum needs.

Because of size and open spaces, some SD schools are located more than 400 miles from the TIE Office. Distance and human resource limitations require TIE staff to be creative and diligent in getting relevant materials and information into teachers' hands.

Resource center staff perceive the center as a key to unlock many opportunities for teachers in rural settings. For example, teachers discover that borrowing software from the resource center can become a strategy for demonstrating to and selling administrators on specific software, a strategy for developing their own computer literacy and evaluation skills, or a basis for building strong, logical arguments for specific computer hardware and software acquisitions. Teachers discover that software integration is a part of a larger process that includes needs assessment, exploration, review, selection, acquisition, integration, and follow-up.

Belridge School District: Planning for Technology Integration
DACOTT 21/20 (District and Community of Tomorrow Today)
A Community of Learners' Model Technology Project

Gary Peterson and Cyndy Everest-Bouch
Belridge School District
McKittrick, CA

Abstract

In the summer of 1987, the Belridge School District, a rural K through 8 one-school district located near Bakersfield, California, posed a critical question. Are we doing enough to prepare children for a future of rapid social and technological change? The governing board realized that education systems of the next century would be different from those of today and determined that to best prepare students for life in the "information age," current educational practices needed to be transformed, and that one of the main components of this transformation should involve the integration of technology. Throughout the fall of 1987, the board, administration, staff, and community developed a vision of what they wanted their school to become. As one would expect among any group of people, there were varying views and concerns about the direction the change process would take. There was, however, general consensus on what was needed:

- A commitment to maintaining a learning environment that is challenging, exploratory, fun, cooperative, and that draws upon a variety of teaching methods and resources.
- A respect for and reliance upon a teaching staff as key implementors of the educational program and a commitment to provide them with numerous opportunities for professional and personal growth in return for their willingness to trust, accept change, risk, and extend themselves.
- A policy that makes the physical plant, staff, and administration available to the community, welcoming community members both as contributors and *learners*.
- A reliance on a governing board that embraces change as a fact of life, believes in the DACOTT 21/20 educational vision, and allocates the resources and materials necessary to make that vision a reality.
- An effort to develop and maintain partnerships, both with business and educational groups, who share these beliefs and interests and have made a commitment to improving the education level of our society.

The vision was to combine all of these ingredients with the most effective, up-to-date learning tools available and with the support and resources necessary to develop and implement a prototype "Community of Learners" Model Technology Project.

The innovative DACOTT 21/20 project was implemented from the ground up. Teachers, office staff, and families each received an Apple IIGS or Macintosh, plus an Imagewriter II printer in the spring of 1988. Software was also provided with each computer. During the remaining school year, the district provided inservice for staff and community to acquaint all with the "friendliness" of their computer. During the summer, extensive physical preparations to the facility and classrooms were completed. In September, 1988, students returned to school to find an Apple IIGS at each of their desks. To insure timely access to the computers by staff and students, all were connected to a local area network which is supported by five Macintosh file servers. Other technological components of the project that were introduced in the fall included a multimedia center, an applied technology lab, an ESC self-paced learning lab, and an "electronic" music room. The telecommunication component includes both satellite and ITFS microwave dish reception, modems for each staff member and family household, online communication with other schools both nationally and internationally, and an FCC licensed low power transmitter that allows for UHF television broadcasting to each home in the district from the school site.

The Belridge School District recognizes and accepts it's responsibility to address the challenge of education for the 1990s and beyond: to return to society an informed, responsible, and more importantly—employable—citizenry.

While the DACOTT 21/20 project represents one of the most technologically saturated sites in the country today, our presentation is not about lots of hardware. Rather, it is a session that personalizes technology; emphasizes shared experiences; focuses on people and change environments; and raises some provocative questions as to how we should look at schooling, staff development, and the curriculum.

Self-Made Database Projects for the Secondary School

Donald Pratt
Bloomsburg University

Abstract

This project shows how locally-made computer databases can put new life into the school curriculum. Some of the not-so-obvious advantages of self-made databases include (1) facilitating meaningful group work, (2) promoting high levels of cognitive functioning, (3) encouraging creativity, (4) bring down abstraction levels, thereby making a clear understanding more immediately available for all, (5) working with real data, (6) discovering patterns never known before, and (7) attaining affective educational and personal goals. The session shows how learning can be spiraled, graphs generated, and a database built from scratch.

Dr. Pratt will show examples of how simple sources, such as the *World Almanac*, can be used to generate very information-rich and high interest projects that can be fueled by the students own interests. By extension, students could also tap libraries and remote databases. Examples appropriate for grades five through twelve will be given for science, mathematics and social studies. This will include self-made databases in water quality, planetary data, and state statistics (population, altitudes, latitude, longitude, highest buildings, etc.). A method of converting the database to graphical form will be alluded to and samples of outcomes will be shown. The effectiveness of the final graphs should be apparent to all.

This session parallels Dr. Mary Harris' session entitled "Self-Made Database Projects for University Level Teacher Education" (W3-6).

W1-7 HIGHER EDUCATION APPLICATIONS (PROJECTS)

Equal Time for Teachers: A Helping of Apple's Pie

Cathy Thurston
University of Illinois

Abstract

As many schools have discovered, it's one thing to have computers—and quite another to use them. A key to the success or failure of computer integration in a given building is its staff development program. When a school receives a major influx of technology, what is the most effective way to train the teachers?

This case study documents the staff development that occurred during the first 6 months of an Apple Equal Time Grant at a magnet middle school in a midwestern city. The study, conducted from July, 1989 to December, 1989, analyzes the various strategies used for teacher training in a school that received a $126,000 grant for integrating computer use and telecommunications into the math and science curriculum. The grant included 22 Macintosh computers, two modems, a scanner, a laser printer, and other hardware and software.

This presentation focuses on what worked and what didn't, including the teachers' own policy recommendations for a school attempting to integrate computer technology with curriculum. The research documents teachers' participation in various training interventions along with their use of FrEdMail, an electronic network. Additionally, it profiles selected teachers' perceptions of the impact of this technology infusion.

To provide a broader context within which to interpret these findings, the researcher sent a questionnaire to all of the project directors of Apple's Equal Time Grants for the past 4 years. The survey was conducted first on AppleLink; subsequently, a U.S. mailing was sent to those people who did not respond by electronic mail. The questionnaire dealt with general demographic information and specific questions related to teacher training and the implementation of the grant at each site. A summary of the responses will be part of this presentation.

Successful Staff Development

Michael I. Tempel
Logo Computer Systems Inc.

Abstract

This session will report on a staff development model that has emerged out of experiences in many schools and districts throughout the United States and abroad. Our work has been with Logo, but the approach may be applied more generally to other open-ended, general purpose uses of technology in which teacher involvement is essential.

Teachers are introduced to Logo in an intensive session of one to three weeks. Short follow-up workshops and site visits are scheduled throughout the school year. At first the focus is on the teachers' own learning and only after that on how Logo will be absorbed into, and possibly change, the learning culture of the school.

The staff development process is permanent. Teachers continue to attend workshops on an ongoing basis. For experienced teachers workshops sometimes focus on technical questions, but most often involve explorations of a subject area or curriculum theme.

How do we measure success? The technology should become invisible and be taken for granted by teachers and their students.

Classroom Applications of Expert Systems

Roy Tamashiro
Webster University

Abstract

The "Institute for the Study of Expert Systems in the Classroom" is a project that develops, evaluates, and documents the applications of expert systems in elementary and secondary schools' curricula. The project explored ways to overcome obstacles that keep expert systems from being widely used in elementary and secondary schools, including their esoteric image, their high cost, and their relatively complex skills necessary to use them.

An easy-to-use expert system, called "Knowledge Works," enabled teachers and pupils (as young as eight years old) to turn their academic and personal topics into workable consultation systems. Instructional applications were developed for grades 2 through 12 in subjects like history, science, consumer education, language arts, literature, and psychology. Pupils also built systems on topics they themselves initiated, such as a restaurant guide, a shopping guide for bicycles, taxonomy of rock music groups, and a system for identifying sports figures.

Since expert systems make it possible to learn about knowledge organization methods (e.g., outline, tree-diagram, networking) through hands-on experience, they are useful in lessons involving *strategic thinking*, (e.g., information organization and problem solving) and *thinking skills* like classification, discrimination, and synthesis.

Middle School Math in a Technological Society

Janet Parker
University of Louisville

Abstract

A primary goal of the Middle School Math With Technology Project is to improve mathematics learning at the middle school level by implementing a curriculum that reflects the impact of technology on both the content and techniques of mathematics teaching. The year-long project is funded by the U.S. Department of Education. After an intensive summer training session, project teachers are developing and piloting units that make regular use of calculators and computers as teaching tools in middle school mathematics, following the guidelines of the recently published NCTM *Curriculum Standards for School Mathematics*. Each teacher has a computer with a projection device in his/her classroom, and a focus of the units is the use of the "electronic blackboard" mode for total class instruction to develop mathematics concepts. Extensive use is made of data analysis type software, including various graphing and spreadsheet programs; no CAI software is used.

The teachers meet four times during the school year to continue the work of the summer component, discuss implementation strategies and problems, and work on revising the units based on their classroom experiences. These meetings are providing critical support for the teachers as they integrate computers into their teaching.

The Right to Succeed—A Progress Report on the Stevens Institute of Technology's Science and Technology Enrichment Project (STEP) for Disadvantaged Students Featuring LEGO TC Logo

Gary S. Stager
SIGLogo (ISTE)

Abstract

The 1989-90 school year marks the second year of the Stevens Institute of Technology LEGO TC Logo Project at Camden Middle School in Newark, New Jersey. This project is a partnership between Stevens Institute of Technology and the Newark Board of Education designed to improve the mathematical and scientific skills in disadvantaged minority middle school students.

The STEP project has placed 18 Apple IIgs systems, each with a LEGO TC Logo interface and construction materials, in four 7th- and 8th-grade math/science classrooms.

The project's first year demonstrated that students' visions of themselves as capable learners were enhanced through constant access to educationally rich materials as LEGO TC Logo, *LogoWriter*, MBL probes, and computers. Early experience caused us to focus more attention on curriculum and staff development. New staff development objectives include regularly scheduled curriculum-based workshops for the math/science and social studies/language arts teachers. Additionally, "Saturday Sandbox" workshops designed to address collaborative problem solving skills, curriculum coordination, and cooperative learning strategies are scheduled for the entire staff. In-school tutors continue to support the teacher's day-to-day classroom efforts.

Our goal for this year is "More Construction...Less Instruction." During the first year, the fear of doing something wrong caused many of the teachers and tutors to teach the LEGO instead of encouraging the students to invent on their own. At NECC, I will share examples of student work and curriculum ideas you may use in your own classroom.

An Experiment with Peer Tutoring in a Microcomputer Applications Course

Janet Smith
University of Tennessee at Chattanooga

Abstract

The diversity of backgrounds and preparation of students in a two-course sequence in microcomputer applications presented an opportunity to experiment with a peer tutoring program. The better prepared students who were doing well at the beginning of the semester could earn extra credit tutoring students who were having problems. Two purposes would be served: being a peer tutor gives the better prepared students who tend to be bored an extra challenge, and having a peer tutor gives the struggling students the one-on-one attention they need.

In the Fall of 1989, twelve tutor/student pairs, matched by days and times available, worked together. The main activity of the tutoring sessions was the tutor helping the student get homework assignments done. Improvements in test performance were moderate in some cases. The expected problem of not having enough tutors did not materialize. The problem that did occur was the students needing help showing a general lack of interest in the program.

Curriculum and Philosophy of the Master Degree in Computers in Education at Inter American University of Puerto Rico

Eduardo Rivera
Inter American University of Puerto Rico

Abstract

It is presented the needs that originated this master degree, how we define the profile, and how we approach the definition of curriculum.

A nontraditional philosophy resulted from this approach, which leads to consider from very practical programming skills to management and leadership roles.

Three assumptions were made: Artificial Intelligence will play a major role in the design of tools and courseware, so as the understanding of knowledge and learning in education, the need to adapt or modify to specific situations and examples of educational courseware which usually comes closed, the need for an active participation of teachers in this adaptation and perhaps some of personalized education with the use of interactive, multimedia, teleprocessing, and intelligent tools.

W1-9 A PROJECT-ORIENTED COURSE IN SOFTWARE ENGINEERING (SPECIAL)

A Project-Oriented Course in Software Engineering

Gary Ford
Carnegie Mellon University

Abstract

Professional software engineering entails a wide variety of activities other than programming. A university course in software engineering that permits students to experience all of these activities can be substantially more educational and rewarding than a course that only allows collaborative programming. This session presents a detailed description of such a course and suggestions for how it can be taught.

The fundamental basis for the course is that is should provide a team project experience patterned after professional software engineering teams. On such teams, different persons play different roles, and very few of the members write code. Therefore, the team is composed of students playing roles such as principal architect, project administrator, configuration manager, quality assurance manager, test and evaluation engineer, designer, implementor, documentation specialist, verification and validation engineer, and maintenance engineer. The instructor plays the role of project manager. The deliverables for the project include documents such as a project plan, a configuration management plan, a quality assurance plan, and a test plan, as well as typical software development documents such as a requirements specification, design, code listing, test results, and a user manual.

A concern in the design of such a course is providing comparable workloads for each student throughout the semester. It is clear that the student playing the role of principal architect or designer will have more work early in the project, while students involved in testing or maintenance will have most work late in the project. The course described in this session addresses this concern by providing additional planning and advisory exercises for each student at appropriate times.

The course can be taught either as a new development project or as a maintenance project. In fact, the latter approach is superior in some ways because it can be based on a larger software system and can give more realistic experiences to the students. Both approaches are discussed, including presentation of detailed materials produced by the Software Engineering Institute to support the maintenance approach.

Fairs, Conferences, Contests, and Activities for Kids

Doris M. Carey
University of Colorado at Colorado Springs

Abstracts

The Rocky Mountain Educational Computing Consortium (RMECCO) of the Pikes Peak area of Colorado recently spent some time examining activities for students of the area. Until 1989, the consortium had sponsored an annual programming contest, but some members felt that there were other ways to promote the educational uses of technology with activities that would benefit a wider range of students.

The idea of a conference for kids was presented by Deborah Hill. She suggested that a conference would allow for a wide range of noncompetitive activities, a broad range of age levels, and a good feeling among the kids that they were participating in an adult-level activity. The plans were realized this spring as the first KidTech 2000 took place on a Saturday morning in April.

In reviewing the popular computer education literature, we found several ideas that inspired us. We wanted to know more about the various fairs, art festivals, nontraditional contests, and extra-curricular activities that take place on an annual basis. In this session, we have invited a panel of speakers to discuss and show the results of their efforts. Many had sound advice that we would like to share with participants.

Among the discussants will be: Deborah Hill and Tracy Brown, co-chairs of *KidTech 2000* in Colorado Springs; Harry Tuttle and Jane McCrohan to discuss Central New York's *Student Compute Fair*; industry representatives to discuss their contests; and Ellen Pruitt and Pat Cutlip who will give us advice on planning a successful computer art festival.

Using Modula-2's Module Concept as the Basis of a Team Project in the First Computer Science Course

Peter Isaacson
Terry Scott
Mathematics and Applied Statistics Department
University of Northern Colorado
Greeley, CO 80639
(303) 351-2215
compsci@unc.colorado.edu

Abstract

The module concept of the Modula-2 programming language can be used to enable students in the first computer science course to become involved in a limited team project. This approach helps to emphasize the importance of a group software development effort and lessen the significance of the individual endeavor in the typical first programming course.

The definition and implementation sections of the module concept provided the design framework for the class activity. The definition part allowed the instructor and students to cooperatively design the function definitions of the chosen data structure while the implementation part permitted each student to implement an algorithm which supported their assigned function definition.

The completed module demonstrates the concepts of abstraction and data hiding in addition to the idea of a data structure being composed of a set of function definitions, a set of algorithms, and a storage structure. Therefore, key components of engineering reliable and easily maintainable software are developed early in the computer science curriculum.

With the addition of any required finishing touches, the module was made a part of the standard Modula-2 library. This enhanced library provides increased functionality for all Modula-2 users as well as an unusual permanence of programming done in the first computer science class.

One of the difficult choices in the initial programming class of a computer science curriculum is between the desire to provide practical environments which involve team programming efforts and the necessity of individual endeavors because of the student's newness to programming and the individuality of grading.

The First Programming Course at UNC

At the University of Northern Colorado (UNC), the first two courses in the computer science curriculum, which need to be taken concurrently, are CS 150—Computer Processing and Algorithmic Design and CS 160—Structured Programming Languages I. The former course focuses on the concepts of data structures, control structures, and common algorithms in a non-language specific manner while the latter course reinforces these concepts in a language specific way.

In the past, the efforts to provide a reasonable environment in the initial course had centered around the use of the separate compilation facility of Pascal-2 by Oregon Software. This enhancement to the original description of Pascal provided a means whereby the students could be asked to write one or more routines which when separately compiled could be linked with the already separately compiled main program which served to test the routines.

For the past three years, the language of choice at UNC for Structured Programming Languages I has been Modula-2. This progression from Pascal was initiated to provide a more complete initial language and to promote better software development practice. Modula-2 has the benefits of Pascal plus the added advantages of the module concept, a more systematic syntax, the process concept, and low-level facilities.

Modula-2 Concepts

With the change to the Modula-2 programming language, the module concept could be used to provide a software development environment which better facilitated a group project. The split of a module into definition and implementation parts provided the opportunity to separate the design from the implementation.

The Modula-2 language has two basic types of modules: a program or main module and external modules. External modules are composed of two divisions: a definition part and an implementation part. The definition portion of an external module contains an export list and the declarations of the exported objects. The implementation portion of an external module contains the actual implementation for the corresponding definition's declarations as well as any local declarations and implementations which are needed.

Since the definition section contains all the information which the importing module needs to know to be able to use the items it imports, the implementation details can be hidden in the corresponding implementation section. Therefore, the exporting module need only make public the interface contained in its definition segment. Because of the separation of these two components, the implementation can be changed without necessitating changes in the calling module provided that the interface doesn't change.

Both the main module and the external modules are separately compiled with full type compatibility checking being done among all modules. In practice, the definition and implementation sections of the external module are often compiled separately. When compiled, the definition portion provides symbol table information for subsequent compilation of its corresponding implementation portion as well as for any other module which imports some or all of its exportable objects.

Another important concept of Modula-2, which enhances its software engineering facilities, is the opaque type. Since its type identifier or name is the only part of its declaration present in the definition, its implementation is not visible to other modules which import and use it. This data hiding allows the importer to declare data objects of this opaque type but not to be able to manipulate them except via the provided importable routines which operate on objects of this type.

Student Background

By the last few weeks of the semester, the students in UNC's Structured Programming Languages I course have been exposed to all major data and control structures in Modula-2. The data structures include arrays, records, pointers, dynamic data types, and opaque types. The control structures include procedures, functions, and external modules.

With this background, the students are prepared to engage in a limited group project. Rather than launch into small group projects which take a semester or more, a class project with the instructor acting as project manager was chosen. This enabled much of the initial requirements and high-level design to be planned by the instructor but discussed and modified by agreement in class.

The Strings Project

The specific class project topic discussed in the rest of this paper is the string data structure. Although it is a common example in many texts, the version of Modula-2 (DEC's WRL Modula-2) used at the time did not have a full string package. Therefore, developing an external module which implemented the string data structure was a practical choice. Additionally, many students have experience with the string data type from high school BASIC classes. (An example portion of the Strings module, without documentation, can be found in Appendix 1 of this paper.)

Function Definitions

Although discussed with the students in the class, a deliberate choice was made to make the string data type an opaque type in Modula-2. This choice reinforced the concept of data hiding and provided an chance to work with a dynamic data type. Modula-2 requires that opaque types be represented by a one-word entity which essentially restricts opaque types to be pointer types. Thus the determination of the storage structure of the string data type could be left to the implementation.

Student participation in the initial stages of the development cycle increased with the set of function definitions for the string data structure. Besides the usual string operations of input, output, comparison, length determination, concatenation, search, deletion, insertion, and extraction, more unusual operations of center, case change, multiple character conversions, decryption, encryption, and pattern repetition were decided on.

Due to the large group nature of the project, no concerted attempt was made to determine a minimal set of operations which would have avoided the overlapping of some of the functions. If the proposed function had some utility in and of itself, it was included in the final set of functions. Of course, this criteria would not necessarily be appropriate for the same topic for a small group project.

Students with a background in any of the many forms of BASIC, willingly contributed their favorite and/or most obscure string operation. For example, the multiple character conversions came from DEC's BASIC-PLUS and includes parity bit trimming, leading and trailing space/tab removal, and case conversion.

Once the set of functions were decided on, the students were assigned a particular function and the task of determining its parameter interface. These were placed on the computer system, collected together, and discussed in class for consistency of user interpretation when using these string operations. For example, it was decided that string positions would be numbered from one through the string length rather than starting at zero.

With the function declarations determined, the definition portion of the string external module was completed. Besides the export list of the opaque string type and its associated set of functions and the function declaration statements themselves, the only other item included in the definition section was an import of the opaque File type from the io library module. This type needed to be imported because it was used as the parameter type for a parameter in couple of the routines.

Storage Structure

The next collective task involved the storage structure and, therefore, the type description of the opaque string type. This decision provided an opportunity to talk about several of the ways in which strings are typically stored in various languages like Modula-2. Both a fixed string length (i.e., a fixed sized array of character) and a variable string length (e.g., automatic dynamic allocation of the needed length as in most BASICs) were investigated.

A related topic of storing of the string's length was introduced. The header method of reserving a certain number of leading character positions (e.g., for encoding the length) or the trailer method of using a following character (e.g., a null character) were mentioned as possibilities.

The combined decision for string length and length storage resulted in choosing a variable length string with an encoded header length. With two leading bytes reserved for the string length, strings of length zero through 65,535 could be accommodated. Also, although the specific size of two bytes for length was compiled into the module, careful attention was taken to allow this to be easily changed via a constant called BytesForLength. Since this constant was found only in the implementation section, the definition section need never be involved in a future change of maximum length.

In DEC's WRL version of Modula-2, the string declaration thus became

```
String = POINTER @nocheck TO StringArray;
```

with a stringarray declaration of

```
StringArray = ARRAY [0..MAXINT] OF CHAR;.
```

This two part declaration with the attribute nocheck allowed strings of all lengths to be compatible. Also, header length and the string positions could simply be treated as elements of a character array of known but variable length.

Algorithms

With the storage structure of the opaque string type determined, the next part of the class project was to determine, design, code, and test the common internal routines which created and destroyed strings. The routines CreateString, DeleteString, and the diagnostic DumpString were need by many of the routines exported by the strings module.

The collective effort and discussion of these routines served to clarify and strengthen the understanding of the implementation of the string type and how to manipulate it. For example, the relationship between the user's view of string positions ranging from one to string length and the implementer's view of the string contents contained in array positions BytesForLength to BytesForLength + Length(String) - 1 (the array subscripts started at 0) became apparent.

Modula-2's dynamic memory allocation routines ALLOCATE and DEALLOCATE were imported from the Storage module of the accompanying library. Due to the bit unit of allocation, the connection between bytes and bits was reinforced. An appropriate constant BitsPerByte was included along with the BytesForLength constant. However, the major use of BitsPerByte was confined to the CreateString routine.

Additional implementation constants of StringMax (in this case 65535), the currently longest representable string, and Radix (in this case 256), the base for the encoding of the string length, were included. The Radix constant was needed since the string type was implemented as a character array and the numeric value of the length needed to be placed in multiple character positions of that array. These constants needed to be computed by hand for inclusion due to the lack of an exponentiation operator to be used in Modula-2's constant expressions. For this situation, the StringMax constant is Radix ^ BytesForLength - 1 , while the Radix constant is 2 ^ BitsPerByte.

Now it was finally time for the students to finish the design of their individual routine and then code and test it. With the complexity of the routine roughly matched to the capability of the student, the relative amount of time to finish the routines was reasonable and workable.

During this later phase of the group project, individual routines were collected from individual accounts and combined into one implementation file. Together then with the previously merged definition file, all routines in their present form could be accessed.

Some Interesting Results

Many interesting and insightful things occurred during the group project. One of the more useful ideas came from a seemingly struggling student. While being tutored by another class member of much more demonstrated ability, he suggested that all routines could be made available as both procedures and functions with a minimum of effort. Simply, write the function, use a call to that function as the procedure body, and assign the return value to the additional procedure parameter. A convention of suffixing the routine identifier with a capital "P" to signify the procedure was accepted and implemented by all class members.

Other enterprising students determined that their particular routine could benefit not only from the common internal routines but also from the other exported routines their fellow classmates were developing. One student managed to use three such routines. Although this approach runs the risk of endless mutual recursion, encouraging creative approaches and using available routines took precedence in this class project.

A very rewarding development the following term was that one of the more capable students decided to take upon himself the completion (i.e., the finishing touches) of the strings module as a non-credit project. Access to the combined definition and implementation files was made available and much productive work was completed. This terminated in the placing of the polished and retested routines in the Modula-2 library directory for access by any user of the system.

In spite of the fact that some students may have been overwhelmed by the size of the overall project, this effect was lessened by the instructor providing guidance and each student's somewhat limited role. Overall this project provided an excellent introduction to valuable software development concepts.

References

[1] J. Mack Adams, Philippe J. Gabrini, and Barry L. Kurtz, *An Introduction to Computer Science with Modula-2*, D. C. Heath and Company, 1988.

[2] Gary A. Ford and Richard S. Weiner, *Modula-2: A Software Development Approach*, John Wiley & Sons, 1985.

[3] Lewis J. Pinson, Richard F. Sincovec, and Richard S. Weiner, *A First Course in Computer Science with Modula-2*, John Wiley & Sons, 1987.

[4] Edward M. Reingold and Wilfred J. Hansen, *Data Structures in Pascal*, Little, Brown and Company, 1986.

[5] Niklaus Wirth, *Programming in Modula-2 (2nd Edition)*, Springer-Verlag, 1983.

Appendix 1A. A Limited Definition Module

```
DEFINITION MODULE Strings;

   FROM io IMPORT
      (* TYPE *) File;

   EXPORT QUALIFIED

      (* TYPE *) String,
      (* FUNC *) Concat,
      (* PROC *) ConcatP,

   TYPE
      String;

   PROCEDURE Concat(    Str1 : String;
                        Str2 : String):    String;

   PROCEDURE ConcatP(    Str1 : String;
                         Str2 : String;
                     VAR Str3 : String);

   END Strings.
```

Appendix 1B. A Limited Implementation Module

```
IMPLEMENTATION MODULE Strings;

   FROM system IMPORT
      (* CONST *) MAXINT;
   FROM Storage IMPORT
      (* PROC *) ALLOCATE,
   FROM io IMPORT
      (* VAR *) terminal,
      (* PROC *) Writef;

   CONST
      BitsPerByte = 8;
      BytesForLength = 2;
      Radix = 256;  (* 2 ^ BitsPerByte *)
      StringMax = 65535;  (* (Radix ^ BytesForLength) - 1 *)

   TYPE
      String = POINTER @nocheck TO StringArray;
      StringArray = ARRAY [0..MAXINT] OF CHAR;

(*  Beginning of example of non-exported procedures/functions *)

   PROCEDURE CreateString(    Len : CARDINAL):  String;
      VAR
         Temp: String;

      PROCEDURE AssignLength(VAR Str : String;
                                 Len : CARDINAL);
         VAR
            Index: CARDINAL;
         BEGIN  (* AssignLength *)
            FOR Index := BytesForLength-1 TO 0 BY -1 DO
               Str^[Index] := VAL(CHAR, Len MOD Radix);
               Len := Len DIV Radix;
            END  (* FOR *)
         END AssignLength;
```

Appendix 1B (continued).

```
      BEGIN  (* CreateString *)
         IF (BytesForLength + Len) > StringMax THEN
            Writef(terminal,
                   "\nRequested string length is too long\n");
            HALT
         END;
         ALLOCATE(Temp, BitsPerByte * (BytesForLength + Len));
         IF Temp = NIL THEN
            Writef(terminal,
                   "\nUnable to allocate sufficient memory\n");
            HALT
         END;  (* IF *)
         AssignLength(Temp, Len);
         RETURN Temp
      END CreateString;

(*  End of example of non-exported procedures/functions  *)

(*  Beginning of example of exported procedures/functions  *)

   PROCEDURE Concat(    Str1 : String;
                        Str2 : String):  String;
      VAR
         TotalString: CARDINAL;
         Start: CARDINAL;
         index: CARDINAL;
         Str3 : String;
      BEGIN (* Concat *)
         TotalString := Length(Str1) + Length(Str2);
         IF TotalString <= StringMax THEN
            Str3 := CreateString(TotalString);
         ELSE
            TotalString := StringMax;
            Str3 := CreateString(StringMax);
         END (* If *);
         Start := BytesForLength;
         FOR index := BytesForLength TO
                      BytesForLength + Length(Str1) - 1 DO
            Str3^[Start] := Str1^[index];
            INC(Start);
         END (* for *);
         FOR index := BytesForLength TO BytesForLength +
                      (TotalString - Length(Str1)) - 1 DO
            Str3^[Start] := Str2^[index];
            INC(Start);
         END (* for *);
         RETURN Str3
      END Concat;

   PROCEDURE ConcatP(    Str1 : String;
                         Str2 : String;
                     VAR Str3 : String);
      BEGIN  (* ConcatP *)
         Str3 := Concat(Str1,Str2)
      END ConcatP;

(*  End of example of exported procedures/functions  *)

   BEGIN  (* Strings *)

   END Strings.
```

Learning to Program with Structure Editing: An Update and Some Replications[*]

Dennis R. Goldenson
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-2173
drg@cs.cmu.edu

Abstract

A previous study showed striking performance differences between a group of students using a structure editor based programming environment and a comparison group using separate text editing and compilation tools. Initial replications showed that students generally prefer the GENIE structure editing environment over other Macintosh software, edit structurally instead of textually when they have a choice, and appreciate and make good use of the structure editing functionality. However the earlier "bottom line" performance differences on exams and programming assignments were not repeated.

The null performance findings can be attributed to curriculum effects. A basic justification for structure editing is that it enables teachers to change the order, pace and emphasis of their courses. But teachers are reluctant to cover unduly demanding materials. In addition it is difficult to change long standing curriculum structure, especially where faculty disagree and/or rely on student assistants unfamiliar/ unsympathetic with the changes.

This paper reports the results of two subsequent replications. When the software is used in more demanding curricula GENIE users again perform substantially better on exams and programming assignments than students using other Macintosh programming environments.

Introduction: Structure Editing and Student Performance

Learning to program in a high level language is a needlessly difficult experience for many of our students. All too often they emerge from a first course ill prepared for subsequent work. The programs that they write tend to be artificially small and devoid of real world complexity. Essential issues of structure and abstraction are deferred until (too) late in the course and/or to more advanced courses. Many students learn that computer science is "too technical" and a field that they can't understand. The more technically adept "detail people" who survive can in fact negotiate through clumsy editors and operating systems, but fall apart when confronted by more challenging program design and comprehension tasks.

A basic rationale behind structure editing is that early and continuing emphasis in a first course can and should be directed towards higher level abstraction and design. One reason that introductory courses tend to dwell on low level aspects of language syntax and tool invocation is that such details are in fact difficult to learn, especially when students do not yet have sufficient context in which to integrate them. But students who use structure editor based programming environments need not spend inordinate amounts of time on tedious and error prone details, since the environments handle them automatically.

Rather than manipulating individual ASCII characters, a structure editor acts on program elements (definitions, declarations, statements and expressions) whose placement is determined by a language grammar that defines legal operators and syntactic classes.[1,2,3] Thus it is impossible to make a syntax error when editing structurally. In addition, since they maintain considerable information about program structure and state, structure editor based programming environments also can provide powerful tools that support a methodical, top-down approach to the design, implementation, testing and comprehension of more complex programs.[4,5,6]

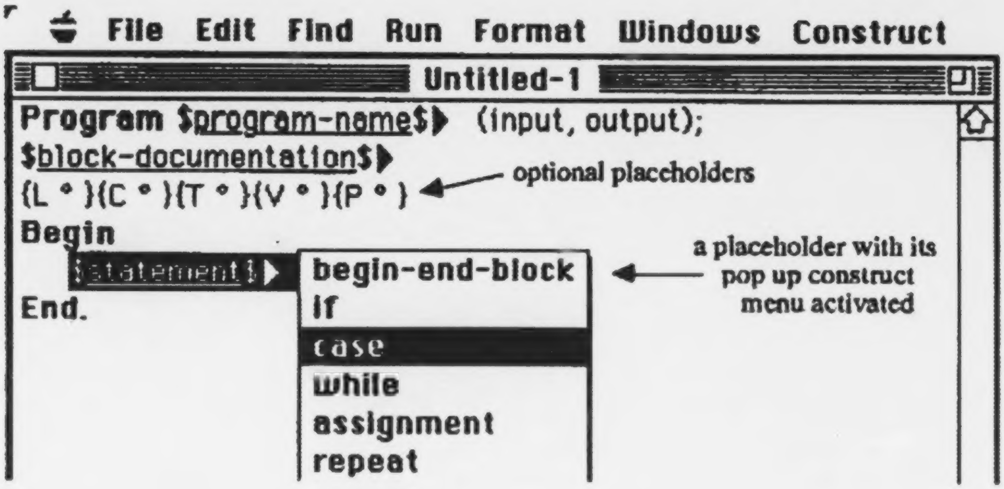
However structure editing also has its down side. It is difficult to create a user interface that is graceful and intuitive to novice users. It can be difficult to negotiate through program structure and invoke basic editing functionality, even in systems with a graphics display or pointing device.[7] Structure editors also can become

---

[*] This material is based on work supported by the National Science Foundation under grant number MDR-8652015. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the foundation.

Thanks to the MacGNOME project group at Carnegie Mellon University: Rob Chandhok, Glenn Meter, Phil Miller, John Pane, Jacobo Carrasquel, Terry Gill, Paul McCartney, Brad Myers, Jim Roberts and Ed Skwarecki. Thanks also to Jeff Bonar, Becky Clark, Larry Faulk, Laura Gardner, David Garlan, Mark Shermis, Bing Wang and John Werth.

Special thanks are due to the teachers who have participated in the studies reported here: Paul Beatty, Jane Bruemmer, Suzy Gallagher, Jim Gillam, Joe Kmoch, David Loewi, Bruce McClellan, Cherise Phillips, David Platt, Laurie Werth and Xiolin Zang.



```
Program $program-name$▸ (input, output);
$block-documentation$▸
{L ° }{C ° }{T ° }{V ° }{P ° }
Begin
    $statement$▸
End.
```

optional placeholders

a placeholder with its pop up construct menu activated

Figure 1. Structure Editing in a GENIE Pascal Program

quite clumsy and bothersome to use, especially as the student becomes more adept and/or the concrete syntax is simple compared to the grammar's underlying abstract structure. The environments remain slow compared to commonly used compilers. Moreover there is the concern that structure editing wrongly discounts the importance of attention to detail.

How well have structure editors in fact fared in classroom use? While it is easy to find strongly expressed opinions, there is little systematic empirical evidence. Early studies have found somewhat mixed results. Scheftic and Goldenson[8] found that a group of teachers and school administrators learning on a structure editing environment consistently performed better on a series of exam tasks than did a comparison group. In addition the structure editing group tended to ask different sorts of questions, focusing on problem substance rather than language syntax. However Chin, et. al.[9] found that users of another structure editing environment performed no better than users of a batch mainframe system on their exams and programming assignments, and that the students preferred the batch environment when offered a choice. Winkler[10] reports some brief experiments where students using a third structure editor did indeed complete better programs in a shorter period of time than students using a standard text editor; yet the latter groups were much more satisfied with their environment.

What can account for these varying findings? Beyond trying to demonstrate that structure editing can affect student learning, we need to identify the conditions under which it does and does not work well. It is unreasonable to expect that all structure editors will always perform similarly. Like any software, structure editors can vary widely in the quality of their user interfaces.[11,9,7] Moreover curricula and teachers can interact with the software quite differently. If the curriculum continues to emphasize short, syntactically correct programs a structure editor becomes redundant at best and introduces needless complexity at worst. Lastly, one can expect individual student differences and background to have an impact.[11,12]

GENIE and Student Performance

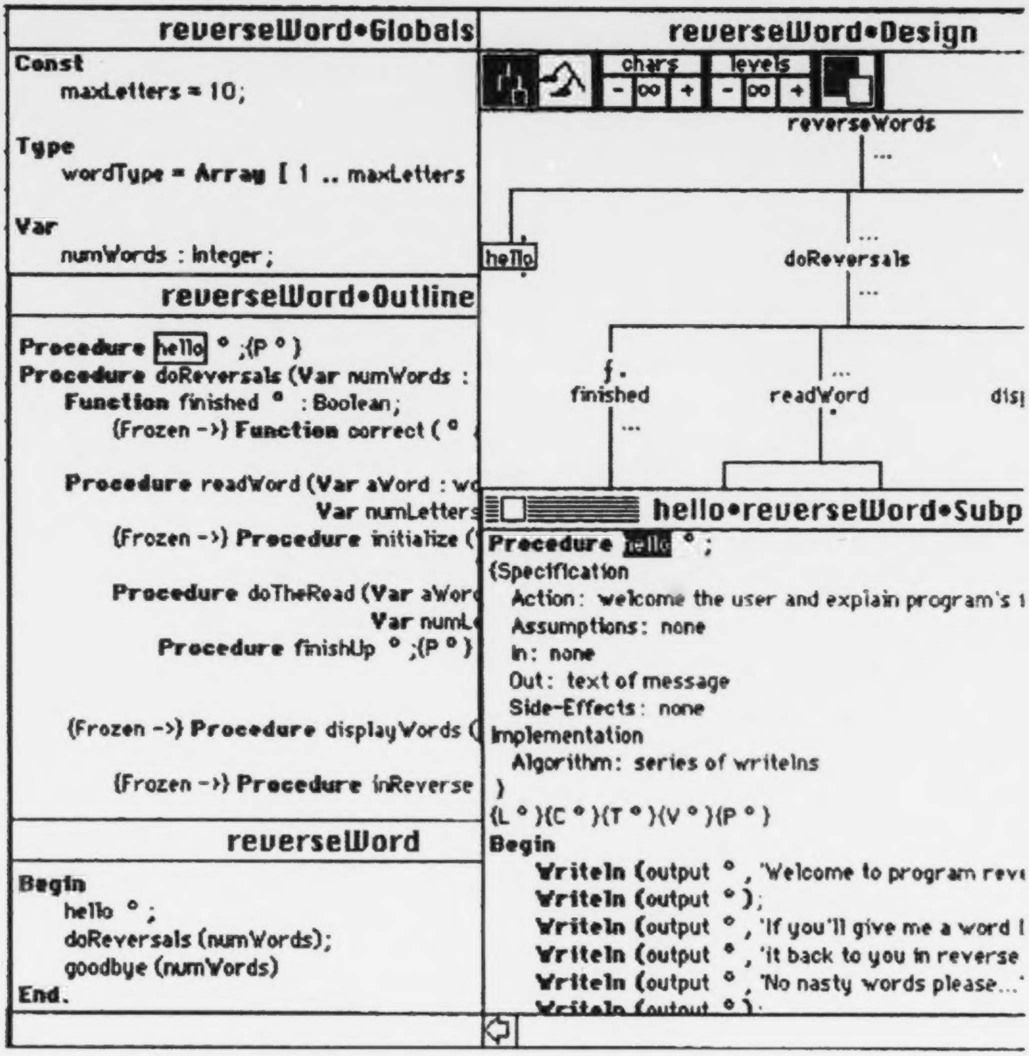
GENIE: The GENIE family[1] of programming environments combines powerful structure editing functionality with the familiar Macintosh user interface.[17] As shown in Figure 1, structure editing is largely a process of replacing placeholders with valid program structure.

The developers have spent a good deal of effort on making structure editing graceful and intuitive to novice users.[7] Perhaps the most common complaint about structure editors is that they unduly constrain the manner in which one can construct and edit a program. So GENIE programs can be edited textually as well as structurally. Text parsing is done interactively, in the fashion of a typical interpreter. Of course doing so introduces the possibility of syntax errors. However, since users routinely move back and forth between text and structure editing, the grain size of most text edits is quite small. While the boundary between text and structure editing remains less "seamless" than we might like, it is fair to say that GENIE environments now do approach the "look and feel" of a good text editor, along with the "intelligence" of a structure editor.[2]

---

[1] Environments are available for Pascal and the Karel the Robot teaching language.[13,14] An Ada GENIE is currently under development.[15,16]

[2] Note though that text editing in the versions of GENIE used for experiments reported in this paper was a good deal less graceful than the GENIE current at this writing. We now are comparing data journaling all user initiated editing events to see whether recent changes in the text-structure editing interface have in fact resulted in fewer editing errors.



```
Const
    maxLetters = 10;

Type
    wordType = Array [ 1 .. maxLetters

Var
    numWords : Integer;
```

```
Procedure hello ° ;{P ° }
Procedure doReversals (Var numWords :
    Function finished °  : Boolean;
        {Frozen ->} Function correct ( °

    Procedure readWord (Var aWord : wo
                        Var numLetters
        {Frozen ->} Procedure initialize (

        Procedure doTheRead (Var aWord
                             Var numL
            Procedure finishUp ° ;{P ° }


    {Frozen ->} Procedure displayWords (

        {Frozen ->} Procedure inReverse
```

```
Begin
    hello ° ;
    doReversals (numWords);
    goodbye (numWords)
End.
```

```
Procedure hello ° ;
{Specification
  Action: welcome the user and explain program's 
  Assumptions: none
  In: none
  Out: text of message
  Side-Effects: none
Implementation
  Algorithm: series of writelns
}
{L ° }{C ° }{T ° }{V ° }{P ° }
Begin
    Writeln (output ° , 'Welcome to program reve
    Writeln (output ° );
    Writeln (output ° , 'If you'll give me a word I
    Writeln (output ° , 'it back to you in reverse
    Writeln (output ° , 'No nasty words please...'
    Writeln (output ° );
```

Figure 2. GENIE's Multiple Views

As implied earlier, GENIE contains a number of advanced program design and testing features beyond its standard structure editing functionality.[5,6,2] Of quite clumsy and bothersome to use, especially as theparticular note given the curricula used in the courses described below are the environment's multiple views of the program database. As shown in Figure 2, the program can be displayed in ways not constrained by the language's concrete syntax. For example the program's call hierarchy can be sketched out in a graphical "design" view, while implementation details can be fleshed out in separate "scope" views for each subprogram. Changes made in any one view are immediately reflected in the others, since all refer to the same underlying program.

The Earlier Studies: In our first semester long field experimental study[18] we compared two college (not CMU) student groups, one of which used GENIE while the other used ULTRIX and VI on networked IBM PCs. The structure editing group performed a full letter grade better on average on both their final programs and examinations.[3] Notably, the GENIE group also performed better than the comparison group on an assignment completed using ULTRIX and VI at semester's end.

This first study can be faulted on several methodological grounds.[18] Perhaps most blatantly, the comparison group used an entirely different hardware and operating system software configuration. Certainly UNIX dialects are in no sense a "straw man". However the comparison group clearly should be as similar as possible to the GENIE group. No single field study is definitive. Hence we are conducting a series of studies to replicate and expand on the initial results.

---

[3] The programs and exams were blind graded.

In studies done at CMU as well as two other universities and several secondary school sites throughout the country students generally do express a stronger preference for GENIE than for other Macintosh software. In addition our surveys show that GENIE students tend to edit structurally instead of textually when they have a choice, and appreciate and make good use of the structure editing functionality.[19]

Following the first study we conducted three additional semester long field experimental studies, one each at another university and at secondary schools in Ohio and Missouri. As just noted, the survey data did continue to show structure editing in a favorable light. However we did not replicate the initial dramatic performance differences. In spite of its apparent potential, software alone seems to have a limited impact.

A basic justification for structure editing is that it enables teachers to change the order, pace and emphasis of their courses. But teachers are reluctant to cover unduly demanding materials. In addition it is difficult to change long standing curriculum structure, especially where faculty disagree and/or rely on student assistants unfamiliar/unsympathetic with the changes.

Both prior to and during these studies it was apparent that the curriculum would have a marked impact on how the software was used. The course at one secondary school was taught by a relatively inexperienced teacher.[4] The other secondary school teacher did have a great deal of experience. However in both instances the course covered only a limited curriculum in Pascal. The university course followed a relatively demanding CS1 curriculum. However it retained a somewhat traditional emphasis on syntax and control abstraction. Most student programs remained small and not particularly complex procedurally.

The Present Studies: We now have completed two additional field experiments.[5] They were done in the same Missouri secondary school used in the first series of replications, and a secondary school in Minnesota.[6]

	GENIE	MacPascal	$p=$*
Final Program			
mean score	73	57	.03
median score	72	56	
N=	22	18	
Final Exam			
mean score	80	68	.04
median score	75	56	
N=	21	16	

* Probabilities are calculated by Student's *t* distribution.

Table 1. Minnesota Performance Differences

In both instances the software was used with demanding curricula more similar to that used in the original study than in the earlier replications. In both instances the GENIE group students again did perform better than those in comparison groups using other Macintosh software.

The same basic research design was followed in each study. One teacher taught both groups of students, used the same curriculum and the same assignments. The comparison classes used MacPascal. As just indicated the curricula were fairly strenuous, with early and sustained emphasis on procedural abstraction as well as comparatively large programming assignments.[7] In addition both schools covered a unit using Karel the Robot prior to moving on to Pascal.[8]

As suggested earlier, the GENIE groups did not perform any better than the comparison group students on the relatively short and straightforward programming assignments given earlier in the semester. However they did in fact perform considerably better on the more difficult and comprehensive work that followed.

Table 1 shows the performance differences among the tenth graders in Minnesota. The GENIE students there did considerably better[9] on both their capstone final programs and their final exams.

The GENIE students appear to have taken good advantage of the software's multiple views of the program database. For example the teacher required that all students turn in charts outlining their programs' call hierarchies prior to completing the programs. The MacPascal students (who parenthetically had previously made good use of Karel's design view) now complained

---

[4] Although she did have prior teaching experience, she had not previously used GENIE nor taught the first course in Pascal.

[5] A study is currently underway in a more demanding course at the same university where we previously found null performance results. In it we have a sufficient number of students to have more confidence that results are not due to prior selection efforts.

[6] Note that the Minnesota school uses a trimester schedule. The course was done in the year's second and third trimesters.

[7] The Missouri high school curriculum was considerably more demanding than what the same teacher had attempted the previous year.

[8] The Missouri students also did a unit in LOGO.

[9] In a few instances they actually did less well.

	GENIE	MacPascal	$p=$*
Program 3			
mean score	81	61	.05
median score	74	68	
residual mean	+3	-3	.07
residual median	+5	0	
Program 4			
mean score	89	60	.02
median score	99	67	
residual mean	+9	-10	.07
residual median	+3	-8	
Program 5			
mean score	89	71	.02
median score†	88	86	
residual mean	+4	-5	.07
residual median	0	-2	
N=	15	13	

* Probabilities are calculated by Student's t distribution.

† Although the program 5 median scores are similar for both groups, the total distributions are shaped quite differently. The GENIE group had considerably fewer low grades. The first quartile for the GENIE group was 83 while it was only 53 for the MacPascal group.

Table 2. Missouri Performance Differences

about the extra work.[10] The GENIE students continued to use the design view to prepare their charts, and to design their solutions more methodically. As Table 1 implies, they also implemented better structured programs.

As most of us try to do, the Minnesota teacher also emphasized the importance of good documentation. Both groups of students received the same exhortations. The GENIE students were more likely to include more comprehensive (and consistently formatted) module comments, using the software's automatically provided subprogram comment templates.[11]

The Missouri eleventh graders using GENIE consistently turned in their assignments earlier and included more functionality in them. Table 2 summarizes the performance of both groups on their last three major programming assignments. Since the GENIE students had done somewhat better as a group in their previous Karel and LOGO work, the table reports a residual analysis as well as the unstandardized differences. Given the small Ns, each residual group difference only approaches statistical significance. However they too consistently favor the GENIE group.[12]

Implications

Structure editing enables teachers to drastically alter the order and emphasis of a first course in programming methods. Since the software environment prevents low level syntax errors teachers can largely ignore such problems, and defer discussion of syntax and formatting details until later in the course when students can better appreciate their subtleties. Since the environment provides well integrated design and program testing tools teachers can better focus attention on more fundamental issues of abstraction and program structure.

Freeing teachers from continual "fire fighting" is particularly important in many high schools and colleges with limited hardware resources, where computer time is all too often very limited. Interestingly enough, the Missouri teacher had been hesitant about trying to cover too much in a secondary school introductory course.[13] The following quotation concisely summarizes both her own experiences and this paper's major theme.

> ...that by the end of the semester I as a teacher was essentially expendable during lab time in the Pascal GENIE class. The students were able to comprehend the error messages and correct their own mistakes for the most part. Many of the MacPascal students grew increasingly frustrated and demanding of my assistance. A major recurring error was unmatched begin/end blocks especially following case statements at the end of procedures resulting in the enigmatic error message "This does not make sense as a statement."

References

[1] Garlan, D., *Flexible Unparsing in a Structure Editing Environment*, Technical Report, Department of Computer Science, Carnegie Mellon University, 1985

[2] Chandhok, R., D. Garlan, D. Goldenson, P. Miller, and M. Tucker, "Programming Environments based on Structure Editing: The GNOME approach," *Proceedings of the 1985 National Computer Conference*, Chicago, 1985.

[3] Teitelbaum, T. and T. Reps, "The Cornell Program Synthesizer: A Syntax Directed Programming Environment," *Communications of the ACM* 24(9), 1981.

---

[10] Note that these students had little if any difficulty making the move to MacPascal. Actually there is some evidence that doing Karel first helps students with their procedural abstraction when they move on to Pacal.[20] If anything, doing Karel first with the GENIE should have muted the differences described in this paper.

[11] The comment templates were not used in Missouri. They are now regularly available through a menu preference item.

[12] Such consistency is much less likely to have occurred by chance than any one comparison taken in isolation.[21]

[13] Indeed both teachers retained throughout the semester a healthy skepticism about the software's impact.

[4] Reiss, S. "Graphical Program Development with PECAN Program Development Systems," *Proceedings of the Software Engineering Symposium on Practical Software development Environments*, ACM-SIGSOFT/SIGPLAN, April 1984.

[5] Roberts, J., J. Pane, M. Stehlik. and J. Carrasquel, "The Design View: A Design Oriented High Level Visual Programming Environment," *Proceedings of the 1988 IEEE Workshop on Visual Language*, Pittsburgh, 1988.

[6] Myers, B., R. Chandhok and A. Sareen, "Automatic Data Visualization for Novice Programmers," *Proceedings of the 1988 IEEE Workshop on Visual Language*, Pittsburgh, 1988.

[7] Goldenson, D., and M. Lewis, "Fine Tuning Selection Semantics in a Structure Editor Based Programming Environment: Some Experimental Results," acmSIGCHI Bulleti, 20 (October 1988).

[8] Scheftic, C. and D. Goldenson, "Teaching Programming Method and Problem Solving: The Role of Programming Environments Based on Structure Editors," *Proceedings of the 1986 National Educational Computing Conference*, AFIPS, San Diego, 1986.

[9] Chin, J., K. Norman and B. Shneiderman, "Subjective User Evaluation of CF Pascal Programming Tools," submitted to *Journal of Behaviour and Information Technology*, 1989.

[10]Winkler, G., "Techniques of User Guidance in Programming Environments," working paper, 1988.

[11]Neal, L. "Cognition-Sensitive Design and User Modeling for Syntax-Directed Editors," *Proceedings of the 1986 Conference on Human Factors in Computing Systems and Graphics Interface*, Toronto, 1987.

[12]Goldenson, D., M. Shermis, L. Werth, L. Flint and D. Loewi, "Programming Environments and Incidental Learning: The Development and Application of a New Computer Anxiety Scale," *Proceedings of the Seventh International Conference on Technology and Education*, Brussels, March 1990.

[13]Chandhok, R., *et. al.*, *Pascal GENIE*, Santa Barbara, California: Kinko's Service Corporation Academic Courseware Exchange, 1988.

[14]Chandhok, R., *et.al.*, *Karel GENIE*, Santa Barbara, California: Kinko's Service Corporation Academic Courseware Exchange, 1987.

[15]Chandhok, R. and T. Gill, "Supporting the Use of Ada in Introductory Computer Science," *Proceedings of 1988 SEI Conference on Software Engineering Education*, Fairfax, Virginia, 1988.

[16]Cleron, M. and S. Fisher, "The Design of the Ada Genie," working paper, August 1989.

[17]Miller, P., and R. Chandhok, "The Design and Implementation of the Pascal Genie," *Proceedings of the 1989 ACM Computer Science Conference (CSC '89)*, Louisville, Kentucky, February, 1989.

[18]Goldenson, D. "Teaching Introductory Programming Methods Using Structure Editing: Some Empirical Results.," *Proceedings of NECC'89*, National Educational Computing Conference, Boston, June 1989.

[19]Goldenson, D., "The Impact of Structure Editing on Introductory Computer Science Education: The Results So Far," *acm/SIGCSE Bulletin*, 21 (September 1989).

[20]Goldenson, D., M. Matchett, B. Wang and M. Chiu, "Programming and Transfer: A Field Experimental Study," poster session, acm/SIGCHI, Seattle, April 1990.

[21]Payne, J. L., "Fishing Expedition Probability: The Statistics of *Post Hoc* Hypothesizing," *Polity*, 7 (Fall 1974).

Logo before Prolog

Alan C. Cutting
Roger Williams College
Bristol, RI

Abstract

Artificial Intelligence (AI) and symbol processing languages associated with AI have received much attention as of late. Most of the teaching material in the language area is directed toward teaching the programming language as an end in itself. In addition most of this material is directed at those with substantial computer experience.

Our goal is to teach AI concepts to students with no computer experience. We use AI programming as a means to promote deeper understanding of AI concepts. Over the past 10 semesters we have had experience with BASIC, Lisp, Logo, and Prolog in this regard. We have found Prolog to be a superior vehicle for illustrating AI concepts in this environment. We have also found that Prolog is most effective if students have had prior exposure to programming and programming concepts such as recursion. A few weeks of exposure to Logo has proved effective in providing this exposure.

This paper relates our experience using BASIC, Lisp, Logo and Prolog. In the accompanying talk we will present and discuss some of the AI programs referred to in the paper as well as programs which illustrate how Logo can be used to prepare students for Prolog. The talk will be directed at those with some experience using Logo. As with the course, no familiarity with Prolog, Lisp, or BASIC is required.

Introduction

Interest in Artificial Intelligence (AI) has exploded over the past 10 years. This growth has been fueled by the broad spectrum of areas touched by AI. Along with commercially viable applications such as expert systems and educational tutoring systems AI holds tantalizing prospects for exploring our own minds and our place in the universe. It's appeal extends to those majoring in philosophy, psychology, education, business, and other non-computer fields as well as those majoring in computer science.

As AI has emerged a number of languages tailored to the symbolic processing needs of AI applications have become available. The choice of a language for use in an AI course for the general undergraduate population is not obvious. By AI course I refer to a course in which the focus is on the concepts and questions associated with Artificial Intelligence as opposed to a course in which the focus is on an AI programming language itself. If the focus is on a language then the choice is easier as a language can be taught to an arbitrary degree in a semester. When the focus is on AI concepts the choice of language becomes significant. Students must be able to master enough of the language so that the programs serve to provide a deeper understanding of the concepts and questions under study. If a program appears too complex to the student it is apt to have the opposite effect, obscuring rather than shedding light on the concept under study. Further, students must be able to achieve this level of proficiency without consuming the semester in the process.

Artificial Intelligence first entered the curriculum at Roger Williams College in 1984. Over the 10 intervening semesters we have taught this topic using BASIC, Lisp, Logo and Prolog. We have also taught the course without any programming content. Based on this experience we have found Prolog to be a superior vehicle for introducing members of the general undergraduate population to AI. The results further indicate that students spending 2 weeks with Logo followed by 8 weeks with Prolog are better able to write and understand AI programs written in Prolog than students spending 10 weeks with Prolog alone.

BASIC

Our first experience with AI was as part of a Computers and Society course. We began by using BASIC to illustrate how computers could be endowed with simple reasoning ability. BASIC was chosen for a number of reasons. The first dealt with our desire to teach AI concepts as opposed to AI programming. At this time the course was aimed primarily at those students with a computer related major or minor. In this setting it was possible to capitalize on the previous programming experience of the students. All the students had at least one semester of programming experience using BASIC. By choosing BASIC we could limit ourselves to teaching techniques and concepts rather than language. The second reason for choosing BASIC was that introductory AI texts at that time were, and for the most part still are, aimed more towards graduate students in Computer Science than sophomores with a computing course or two. An exception was *Experiments in Artificial Intelligence for Small Computers*, by Krutch. Krutch does an admirable job of presenting AI concepts in a simplistic way using BASIC. A third factor was the availability of software. At that time the college computing facilities consisted of a large timesharing system with standard language support (BASIC, C, Cobol, FORTRAN, Pascal, PL/1, RPG, etc.) and a few Apple II microcomputers. Given the apparent suitability of BASIC, searching out and funding an additional language seemed unnecessary.

BASIC, in combination with Krutch's book, proved moderately successful. Two major programs, one which performed simple syllogistic reasoning, and a simple checkers program provided a good basis for discussion. The size and complexity of the programs did however overwhelm many of the students obscuring the general principles. While it was not necessary to teach programming, the amount of time spent explaining the translation from principles to code was significant. Students were unable to implement improved heuristics for the checker program because of the program complexity. Programs dealing with other topics such as natural language were omitted for lack of time. We began to look for alternatives.

Lisp

Common Lisp had recently been introduced for both the Data General timesharing system and the IBM PC. This development in conjunction with our discovery of a more theoretical yet understandable book, *Inside Computer Understanding*, by Schank and Riesbeck prompted us to switch. Lisp however proved to be too much for both our timesharing system and our students.

Common Lisp at this time required about two megabytes of dedicated memory per user on the timesharing system. Our 50 user system quickly became a 4 user system. If a fifth user typed Lisp the 4 user system became a no user system. About the best that can be said of this adventure was that some students, observed wearing buttons sporting the words "HAVE YOU GOT A 2Mb HEAP", did apparently learn a bit about computer architecture and the limits of the same. Fortunately, Lisp had also been recently introduced for the IBM PC. Having just installed a PC classroom at the college we made a hasty switch.

The PC version of Common Lisp proved a joy to use. It's integrated editor with it's parenthesis matching feature was a significant improvement. It also came packaged with the San Marco Lisp Explorer, a computer based tutoring system for Lisp. Even with these tools however it soon became clear that Lisp was itself going to consume the semester. For students trained in BASIC, Cobol, and RPG the concepts of list management and recursion proved difficult. Again we went searching.

Prolog

As Prolog interpreters became available for the PC we began to look in that direction. After evaluating several interpreters we settled on Arity Prolog. The basic problem with Lisp had been that students needed to know a lot of Lisp before they could write or understand simple AI programs. Prolog allows one to dive immediately into AI. While advanced programs are as complicated, perhaps more complicated, in Prolog as they are in Lisp elementary programs are much easier. Within a week students with an understanding of basic programming techniques can write and understand interesting, instructive AI programs. As they learn more about the language they can write and understand more interesting programs. With Prolog we found that our students were able to write and understand the same type of programs we had used with BASIC in a fraction of the time. We could go on to explore natural language, the blocks world, and other topics. Recursion however remained a stumbling block for many students.

Had our audience remained the same we probably would not have looked past Prolog. By this point interest in AI had grown well past those students with computer related majors or minors. In order to serve this wider audience we redesigned the course as a General Education course devoted to the AI topic. The new course is designed for students with no computer experience. The emphasis on AI concepts as opposed to AI programming remained. During the course students are introduced to AI programming as a means to promote deeper understanding of selected AI topics. It was our feeling that students with no computer experience would consume too much time getting started with Prolog and would have considerable difficulty with the recursive aspects of Prolog.

Logo

Logo looked like an attractive alternative. Logo seemed to be a logical choice for introducing programming to students with no computer experience. As a close relative of Lisp it seemed suitable for developing AI applications as well. The LCSI Logo system was easy to use and affordable.

As an introductory language Logo proved to be outstanding. The built in graphics hold the students interest and encourage them to experiment while they learn basic programming concepts. The use of the "flip side" facilitates the modification of procedures, again encouraging experimentation. Within a week students, finding that programs made up of many small procedures are easier to write and debug, "discover" the principle of functional decomposition for themselves. Logo graphics provide a tool which enables students to make real progress with the concept of recursion. Complex geometric shapes drawn with simple recursive procedures bring recursion to life. Students are drawn in to ponder how such interaction is possible. In the process they not only "discover" recursion but are exposed to a new mode of thinking about complexity. Does producing something as complex as human behavior necessarily require a mechanism of comparable complexity?

Logo proved an excellent vehicle for introducing students to programming and programming techniques. When applied to AI problems it suffered the same fate as Lisp. The complexity associated maintaining and manipulating lists obscured the AI concepts we were trying to illustrate.

Logo before Prolog

Our next step was an obvious one. We currently use Logo to acquaint students with programming and concepts such as functional decomposition and recursion. Once students understand recursion we move to Prolog to illustrate AI concepts. In practice we have found that 2-3 weeks of Logo followed by 7-8 weeks of Prolog are sufficient to introduce students to a good range of topics in AI.

In the course of developing an effective language strategy we have also developed a textbook strategy. We have moved away from AI programming language texts altogether. Programming/AI texts generally use AI concepts to illustrate language features. We, in contrast, use programming to illustrate and promote deeper understanding of AI concepts. In their effort to illustrate language features the AI programs in texts frequently include "bells and whistles" not required to illustrate the AI concept at hand. We have found students are generally able to pick up all the Logo and Prolog programming they need in class. For the few students who need extra reinforcement with Prolog we recommend a text such as *A Prolog Primer*, by Jean Rogers, which covers only the language and does so in the simplest possible way. All students seem capable of mastering Logo without a text.

For introducing students to the broader concepts and questions associated with AI we have found that a text is necessary. As was previously mentioned however, most AI texts seem to be written with computer science majors in mind. Their focus tends to be too detailed and they generally assume skills our audience does not have. We have found that a book such as *Machinery of the Mind*, by George Johnson, which discusses the concepts and questions of AI in a historical setting and in a non-technical manner provides a more effective framework for the course. As we move through the book and encounter a discussion of an interesting program associated with a concept we drop into programming mode to examine it more closely.

Conclusion

Interest in AI extends well beyond those majoring in computer science. For this group AI programming is not an end but a means to an end. Their goal is to understand the concepts, questions, and issues associated with AI. Writing and modifying AI programs is a means to that end if the students can master enough of the language to understand the programs without consuming the semester. Logo seems to be a particularly efficient and effective vehicle for teaching the programming concepts necessary to understand Prolog. Prolog's strength is that it allows students to begin doing useful work without extensive study of the language. By using this language combination we have been able to bring students to a point where they can begin writing interesting and useful AI programs in a matter of weeks.

References

Arity Corp. (1986) *The Arity/Prolog Programming Language*. Concord, MA.

Gold Hill Computers, Inc. (1984) Golden Common Lisp. Cambridge, MA.

Johnson, George (1986) *Machinery of the Mind*. Redmond, WA: Tempus.

Krutch, John (1981) *Experiments in Artificial Intelligence for Small Computers*. Indianapolis, IN: H. W. Sams

Logo Computer Systems Inc. (1986) *LogoWriter Reference Guide*. Montreal, Canada.

Rogers, Jean B. (1986) *A Prolog Primer*. Reading, MA: Addison Wesley.

Schank, Roger and Riesbeck, Christopher (1981) *Inside Computer Understanding*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Steele, Guy L. (1984) *Common Lisp, The Language*. Digital Press.

NSF Funding Opportunities

Anita J. La Salle
The American University and The National Science Foundation

Abstract

This session will address National Science Foundation (NSF) funding issues of interest to educators. In particular, the following funding opportunities will be discussed:

- Funding for Science, Engineering, and Mathematics Education:

  Undergraduate—laboratories, faculty enhancement, course and curriculum development, career access for women, minorities and the disabled, and research experiences for undergraduates.

  Pre-College—materials development and informal science education, and teacher preparation and enhancement.

- Funding for Computer & Information Science and Engineering:

  Including infrastructure awards, instrumentation, research initiation, computing research, and cross-disciplinary activities.

- Funding for Targeted Audiences:

  Including research opportunities for women, assistantships for minority students, research assistance for minority scholars and institutions, research awards for predominantly undergraduate institutions, and funding for international cooperative science and engineering activities.

True BASIC as a Language for Computer Science

Thomas E. Kurtz
Dartmouth College
True BASIC, Inc.

Abstract

Pascal is almost universally accepted as the language of choice for elementary computer science. Virtually all such courses are based on it, good textbooks abound, good microcomputer implementations are readily available, and the CEEB Advanced Placement Test presumes it. But the ideas of computer science are language independent. This paper will explore the use of an alternative language–True BASIC–for teaching computer science. True BASIC conforms to the ANSI Standard for BASIC and, like the Standard, includes the usual structured programming constructs. It also includes *modules*, which are a way to package external procedures that can share otherwise hidden data.

We will examine the topics of an elementary computer science course and compare their treatment under Pascal and under True BASIC. These topics will include programming concepts, elementary searching and sorting algorithms, empirical and theoretical performance of algorithms, abstract data structures, data structuring, and pointers and allocated storage. And we will suggest a syllabus for an introductory computer science course using True BASIC.

Using CD-ROM for Research in the Media Center and Classroom

Michael K. Russ and Janet Leistner
Evansville-Vanderburgh School Corporation
Evansville, Indiana

Abstract

In our society today, the most important product we deal with is information. Providing our students with a tool to better handle this tremendous resource is a must. The use of CD-ROM by media specialists and classroom teachers as a research tool is one way to more efficiently deal with this information overload we are experiencing. The presenters will show how CD-ROM can be used by students and teachers from the elementary grades through high school for doing all types of research. Challenges of start-up for CD-ROM, types of discs available, and research strategies are among the topics to be presented. Attendees will receive a copy of the CD-ROM guide being developed for teachers in the Evansville-Vanderburgh School Corporation. The guide will include sample CD-ROM worksheets for use by students and teachers.

Integrating Computer Skills into a Transitional Education Program

Anthony H. Robinson
Putnam County Department of Education
Cookeville, Tennessee

Abstract

This session will include a comprehensive overview of a transition from school to work program for special needs and economically disadvantaged at-risk students. A short video presentation of students on the job and in the classroom will be included. Participants will be able to view materials actually used within the transition program. A question and answer session will be held toward the end of the session. Information regarding the implementation of a transitional education program will also be available.

Team Logo Meets Newton and Maxwell

Camille Minichino and Stephen C. Sesko
Science Education Center*
Lawrence Livermore National Laboratory

Abstract

Team Logo is a group of selected students, all highly proficient in Logo programming, and highly motivated in learning. These girls and boys are unique in that they come from many different school districts to work together for three weeks during the summer, and the age range is very broad (grades 5 through 10). During the four years the group has been together, they have used Logo to investigate challenging and complex ideas in mathematics and science, among which are the Fibonacci numbers, trigonometry, fractals, and chaos. They have also worked on advanced programming ideas such as remodeling "broken" Logo commands, the use of global variables, and recursion.

During the summer of 1989, Team Logo explored concepts of physics. The students were introduced to the four basic forces of physics: gravitational, weak, electromagnetic, and nuclear. They then created a microworld in which these forces could be investigated. The students began their study by giving the turtle first mass, to investigate the gravitational field, and then charge, to investigate the electromagnetic field. As sample phenomena, the students modeled free fall, reflection, and refraction. In each case, the students were required to determine the appropriate constants for their microworld. They were also taught, and modeled, vector arithmetic as a tool for calculation of net forces.

This project in curriculum development was designed by the two presenters, one a computer education specialist (SCS), the other, a physicist (CM). The project was implemented in a cooperative learning environment, where both the students and the mentors studied and learned together, sharing information and solving problems.

The Science Education Center is an educational outreach program of the Lawrence Livermore National Laboratory. The Laboratory is managed by the University of California, under contract to the United States Department of Energy.

W2-5 ELEMENTARY & SECONDARY CLASSROOM APPLICATIONS (PROJECTS)

A Transformed Learning Center: Using Technology to Restructure Schools

Sara O. Schoenefeld
Transformed Learning Center
Jenks, Oklahoma

Abstract

TLC, Transformed Learning Center, provides a technology-rich educational environment for 120 high school freshman. Designed as a system for transforming traditional teaching and learning processes, TLC offers 120 high school students a flexible, student-centered, self-paced environment. Four teachers, one each from the core areas of mathematics, science, social science, and language arts, comprise the multidisciplinary team that will accompany these students through their four high school years. Students and teachers access 25 Memorex-Telex workstations, 14 IBMs and 40 Apple Macintosh computers, laserdisc hardware and software, CD-ROM, and online retrieval capabilities.

A formal research study will compare the 60 male and 60 female TLC students who represent all ability levels, races, and socio-economic conditions to a matched sample in the traditional school environment. The 4-year project and 8-year longitudinal follow-up studies will provide data to compare the impact of the new educational design with the traditional pattern in the areas of academic achievement, higher order thinking skills, school and job satisfaction, self-concept, and preparation for lifelong learning. Project participants expect to demonstrate that the infusion of computers and other interactive technologies used in a fundamentally different way make a significant difference in student and teacher performance.

The Computer-Greenhouse Effect

Alan Solomon
The School District of Philadelphia

Abstract

South Philadelphia High School is located in the city's core and receives Chapter 1 support. The school has a high dropout rate and addressed this problem by incorporating a computer component into its life science program. The school received materials from Apple Computer through a competitive award.

Potential dropouts, students enrolled in special education programs, students with limited proficiency in English and slow learners formed the target group. These students were paired with their advanced and average counterparts in a series of learning experiences that were facilitated by the computer. Specifically, 20 Apple II GS computers were placed in the school's horticulture program where students gathered, recorded and analyzed data, worked through a series of simulation activities, and prepared laboratory reports. Frank's Nursery adopted South Philadelphia and provided staff time and supplies.

At this time, more than 600 students have participated in the program. Our evaluations have revealed that the participants enjoyed the program and would like to expand their computer knowledge and use. Next year. through funds provided by the school district, the laboratory's hours will be extended in order to allow more students to use it. The laboratory will also be opened to the community.

Using Dazzle Draw with Behavior Disordered Students

Sandra Smith
Riveredge Hospital
Forest Park, Illinois

Abstract

Fishing! This produces a mental image of quiet, peaceful days. But what if fishing were compared to teaching children and adolescents with behavior disorders? The teacher is the fisherman and the students are the fish. These are not the run-of-the mill fish but tough ones that have seen many baits or enticements. These fish recognize these lures and reject most of them. The challenge is to find a hook baited with a new approach that will draw the nonacademic fish to the fisherman and subsequently into his boat.

Computers are the hook for behavior disordered students and *Dazzle Draw* is the bait. The innovation is the use of computer drawing programs to allow the student a new freedom of expression of feelings about self and others. *Dazzle Draw* from Broderbund Software is an art program that allows the user to draw directly on the computer monitor with the use of the mouse. Artistic skills are not a prerequisite for working with *Dazzle Draw*. The student is given a blank computer canvas and invited to create. Even primary grade children grasp the concept of the software quickly. Thoughts and feelings are easily transferred to the computer screen as if by magic.

Telecommunications for Critical Inquiry: The Homeless Around the Continent

Lynne Schrum
University of Oregon

Abstract

This project, which was conducted with elementary and middle school classes in the United States, Canada, and the Virgin Islands, was designed to improve civic education through use of a critical inquiry method of investigation among the participants. The students ranged in age from third to seventh graders. For several months these groups examined their local homeless situation, researched solutions, and actively pursued information they deemed important. Students interviewed community members with divergent points of view, participated in simulations, wrote about their experiences, and communicated with the other classes via a telecommunications network. The interaction was enhanced by exchanges of other materials. An integrated approach was used, as the students found they were engaged in math, writing, and science activities. The results of this activity were far reaching and exciting. Ideas for replication or expansion will be shared.

Indiana's Buddy System Project

Brenda Raker
Apple Computer, Inc.

Abstract

During 1987, a project was conceived by a small private sector to test the feasibility of using technology in the home as a catalyst for improving education and increasing economic opportunities within the state of Indiana. This project became known as the Buddy System Project.

While the project's vision is to provide every Indiana student in grades four through twelve access to home-based technology, the initial focus concentrated on the fourth grade. The services that could eventually be available over this home educational network include secondary education, adult education, and primary and vocational offerings.

Project funding and microcomputer hardware and software have been provided to pilot this home technology concept in five elementary schools during the 1988-89 and 1989-90 school years.

Initial feedback indicates that the project has been successful. For example:

- The technology has increased parental involvement.
- Many parents and siblings are involved with and are using the computer.
- Teachers report increased creativity, research ability, time focused on educational topics, and enthusiasm by their students.
- Teachers report that availability and use of the technology has changed their teaching style fundamentally and invigorated their careers.

The next steps involve expanding the project to the sixth grade and increasing the number of sites.

European Schools Project

Henk Sligte
University of Amsterdam

Abstract

The European Schools Project is a research and development project initiated by the University of Amsterdam, designed to enable secondary schools in the 12 member-states of the European Community to explore the possibilities of educational telecommunications for the improvement of their education. As a central application the concept of "teletrip" is used, a combination of local research activities of pupils around a certain topic, supported by various teachers from various disciplines, and global exchanges of results with other sites using electronic mail.

A teletrip is a specific elaboration of what one refers to nowadays as collaborative distance learning. Teletrips stimulate the innovation of regular education in a number of ways. Teachers are involved in a cooperative process of educational design and planning, subject matter is fertilized with real-world knowledge, allowing intercultural and intercontextual dimensions, traditional interaction patterns in classrooms are transformed under the influence of working in cooperative taskgroups and by the shifting roles and functions of the teachers.

The presentation will pay attention to the current situation, like the number of countries, schools, and projects, the anticipated progress and new developments, and the problems concerning the implementation of teletrips in the educational system.

Maryland Education Project: Computer Integrated Instruction Training

Barbara Reeves and Patricia Mullinix
Maryland State Department of Education

Abstract

The Maryland Education Project is an exemplary model of collaboration by business and education to improve learning and instruction through the use of computers. Since 1987, Potomac Edison, a utility company, has donated 89 networked computer labs to six local school systems and eight institutions of higher education in its service area. These school systems, the Maryland State Department of Education (MSDE), and Potomac Edison have built a collaborative operating structure to ensure that the technology will be use wisely and well. Building on the collaboration, MSDE, this past summer, submitted and was awarded a grant from the U.S. Department of Education to provide funds for more extensive training, practice, and feedback to the teachers in the project to help them make the computers a true instructional and learning tool.

A major component of the integration process includes the development of specific training modules to train teachers in integrating technology into the elementary mathematics curriculum. The training modules, to be developed collaboratively by the partners in the project, include strategies for using the spreadsheet, word processing, databases and Logo to help develop problem solving skills.

Chapter 1 and Technology: De-Papering Chapter 1

Jean C. Reynolds
Consultant, Sanford, North Carolina

David H. Berenson
New Hanover County Schools

Abstract

Chapter 1 has advanced to the front through the use of technology. All participants were introduced to a new and different field of technology. With each lab being equipped with computers and each lab teacher having had a substantial amount of computer training, this technological trend has proven to be very effective in the lab setting. Participants used various instructional software programs in the reading and math areas. The first choice of software was MECC, followed by Apple's Early Learning series. The varied instruction possible due to technology provided an unlimited source of curriculum applications for the at-risk student. After using their new-found technology for a year, the participants moved into another phase. Chapter 1 is required by federal law to evaluate its effectiveness each year. What better way for participants to handle this type of evaluation and reporting than through the use of *AppleWorks*, with its integrated applications? Participants created a database and spreadsheet comprehensive enough to handle their gains in CAT scores, needs assessment, transfer slips, enrollment dates, and more. The data can be analyzed from the reports to clearly confirm the efficacy of the Chapter 1 program. This project is valuable for teachers and administrators in Chapter 1 as it includes computer curriculum development, teacher training, and research information. The project is not limited by grade level, providing flexibility for any Chapter 1 program.

Dynamic Writing: A Glimpse into the Future

Ihor Charischak
10 Bogert Avenue
White Plains, New York 10606
(914) 946-5143

Abstract

When we think of writing, we usually associate it with the act of recording letters of the alphabet, in a recognizable pattern, for the purpose of communicating an idea. The traditional tools for writing have been paper, pencils, pens or typewriters. Today there is also word processing, which makes editing easier, and desktop publishing, which makes the results of writing more attractive and dramatic. Still, the basic unit of writing is the printed word. This association of writing with words on paper is so common and taken for granted, that rarely do people think about its being anything else. But let's imagine for a moment that we can gaze into a crystal ball and see what writing might look like in the future, keeping in mind the potential of our current technology. What we might find is a writing environment where we can make words "come alive" by giving them a visual context with graphic displays, motion, and animation. Simple written communications like letters become dynamic events, stories turn into video presentations, and social studies reports give the reader the feeling of being there.

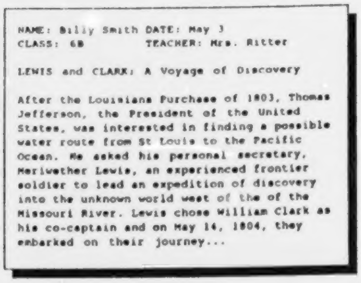
Today there is a piece of software called *LogoWriter*™ that allows a student to "write" in this dynamic way. This paper will take a look at this unique way of writing that may someday be the norm.

Paper

Learning is something that comes naturally to children. By interacting with their environment and imitating the behavior of people around them, they learn a great deal in very short time. It is in the early formative years that another "teacher" enters their lives: television. TV provides a simulated reality in which spoken words have a visual context so the child in a sense "sees" what is being said. Classroom teachers use this idea all the time to help them teach. They point to a place on a globe to help students get a sense of where France is in relation to the rest of the world. Books use a picture of a chipmunk to clarify what a furry rodent in the squirrel family is. But neither of these learning resources is as effective as actually being in France or seeing a chipmunk dashing into its burrow. Since television and movies are only one step removed from reality, they are naturally a more interesting mode of communication for children. Since writing or reading usually does not have a visual context, it is not surprising that many children resist using the written language as a form of communication, especially if they have not experienced the power of words to communicate their ideas.

One of the characteristics of a computer that makes it such a powerful resource is that it can simulate for the student a learning environment that would be otherwise difficult to create. For example, the computer can simulate a writing tool where text can easily be given a visual context so students can "see" what they are writing. For example, let's say Billy (who happens to be in the 6th grade) has a social studies report to write and he is versed in using *Dynamic Writer* (a fictitious name for a computer based writing tool.) Since Billy has just returned from a family vacation to the northwestern states, he decides to write about the explorers, Lewis and Clark, whose journey was highlighted on many historical sign posts in Montana. Here is what the report might look like if it were written in a more traditional way.

LogoWriter is a trademark of Logo Computer Systems Inc.



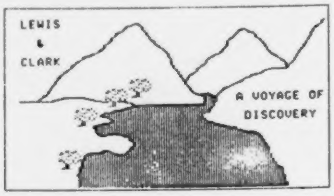
NAME: Billy Smith DATE: May 3
CLASS: 6B TEACHER: Mrs. Ritter

LEWIS and CLARK: A Voyage of Discovery

After the Louisiana Purchase of 1803, Thomas Jefferson, the President of the United States, was interested in finding a possible water route from St Louis to the Pacific Ocean. He asked his personal secretary, Meriwether Lewis, an experienced frontier soldier to lead an expedition of discovery into the unknown world west of the of the Missouri River. Lewis chose William Clark as his co-captain and on May 14, 1804, they embarked on their journey...

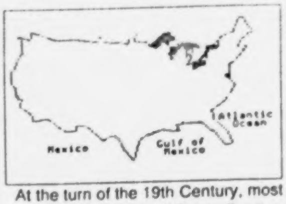
The essence of a report written with *Dynamic Writer* can not be as easily captured on paper. It needs to be "read" on the computer.

First there's the title page:



Billy mixes text and graphics to create the title. *Dynamic Writer* makes it easy for him to draw the river and hills. After a few seconds, the computer displays the next page which is "dynamic." This means that there is a sequence of events.

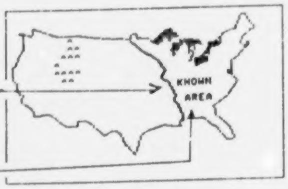
Then a map of the United States appears. ⟶



Next this text appears. ⟶ At the turn of the 19th Century, most citizens of the United States lived east of St. Louis.

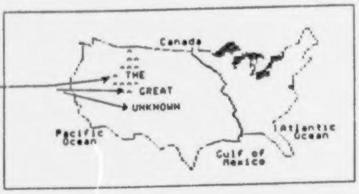
Then a line is drawn ⟶

and the civilized area is noted with "KNOWN AREA".



After a short wait this text appears ⟶ But west of the Mississippi

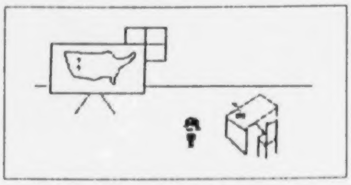
followed by this text blinking off and on ⟶



What made this page dynamic was a sequence of instructions that Billy gave the computer. He directed the events in a way that maximized the visual impact of the communication.

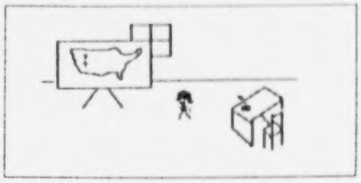
The next page of the story is also dynamic. It includes motion and dialogue.

The scene shifts to President Thomas Jefferson's office.

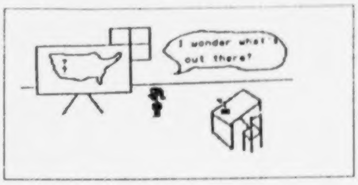


Thomas Jefferson was a curious president. He wanted to know if there was a waterway to the pacific.

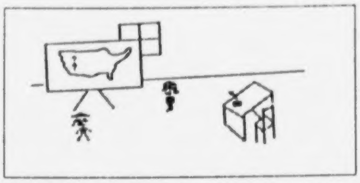
He walks to the map



and thinks out loud.

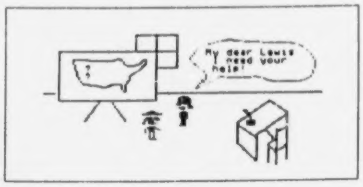

I wonder what's out there?

The balloon erases and new text appears.

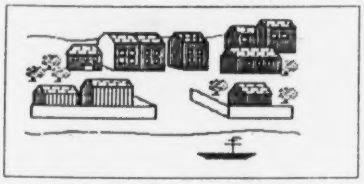


The president beckons his personal secretary and experienced frontiersman, Meriwether Lewis, to lead the expedition.

Lewis walks up to the president. A new balloon appears.

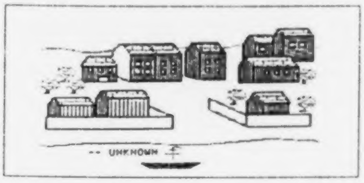

My dear Lewis I need your help!

Lewis accepts and he chooses William Clark as his partner on the expedition. The scene switches to St. Louis. The date is May 14, 1804. Lewis and Clark are about to embark on their perilous trip.

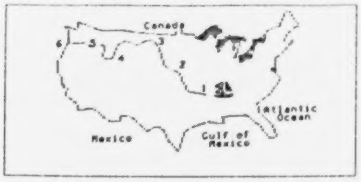


And so on May 14, 1804, Lewis and Clark left from St. Louis on a trip to the...

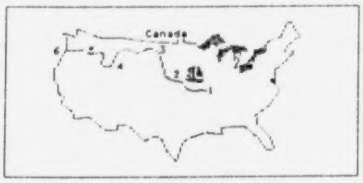
As the boat moves from right to left, the word UNKNOWN blinks like a neon light.



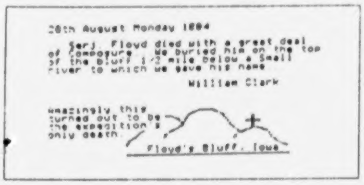
Billy continues by highlighting the major events that occurred during the course of the trip. He used the boat as transitional device to move from one location (and significant event) to another.



When the boat reaches the next point on the river (2),



a page summarizing the significance of that place appears.



20th August Monday 1804

Serj. Floyd died with a great deal of Composure. We buried him on the top of the bluff 1/2 mile below a Small river to which we gave his name.

William Clark

amazingly this turned out to be the expedition's only death.

And so on.

In the future *Dynamic Writer* may be as ubiquitous as AppleWorks is today. But that will depend on how powerful, flexible, and user-friendly the software is. Today there is a computer program that was designed with dynamic writing in mind. It is called *LogoWriter™*. (In fact, Billy's report was written in LogoWriter.) Many schools are presently giving students this unique tool to write with. From my observations of many of these children, it is apparent that they enjoy the writing process more, since they can give their words a visual context. This contrasts with more traditional writing which may make bulletin boards look nice, but is rarely read by others. Because of the engaging qualities of dynamic writing students show much greater interest in not only sharing their own work, but also in reading the work of their classmates.

As technology continue to improve, we will probably see more and more people writing dynamically. This will not only make written communication more interesting, but more students will get turned on to the writing process. Who knows, maybe some day, it will be a common sight to see people curl up under a tree with their favorite dynabook[1] or write animated letters to their friends. *LogoWriter* is certainly a step in that direction.

[1] Dynabook was Alan Kay's name for a computer project that he proposed to his bosses at Xerox's Palo Alto Research Center in 1972. He described the dynabook as a "dynamic media for creative thought." *(taken from Smith & Alexander, Fumbling the Future: How Xerox Invented, Then Ignored, the First Personal Computer (Morrow & Co., New York: 1988), p. 84-85)*

Computer Labs in College Residences: Opportunities for Informal Learning—Especially for Women and Minorities

Pamela Freyd, Director, PennLincs
(215) 898-6478 pam@linc.cis.upenn.edu

Christopher Dennis, Director, College House Program
University of Pennsylvania, 3815 Walnut Street, Philadelphia, PA 19104
(215) 898-5551

Thomas Kinsella
Department of English
Stockton State College, Pomona, NJ 08240

Abstract

This paper reports on a year-long study of three computer labs located in college dormitories at the University of Pennsylvania. Although the University maintains many computer labs on campus for student use, faculty involved with residential life felt that placing computer labs in residences might help extend the educational space of the University (Ziller, 1987). A gift of Macintosh computers from the Apple Corporation made it possible to open three labs in September, 1988. Data collected through surveys, logs, interviews and observations, focused on who used the labs, what the labs were used for, when and how students worked in the labs and what changes seemed to occur over the course of the year. Results support the notion that residential computer labs offer an excellent opportunity to extend the learning space of universities, especially in fostering informal support groups for such areas as writing. Results indicate that these labs offered an important opportunity to provide women and minorities with informal learning experiences that enhance computer expertise, experiences which should help to give them a more equal footing in the college experience.

> *"The very existence of the lab has saved me. I don't know what I would have done this year without the lab."*

Introduction

As a result of a gift from Apple, Macintosh computer labs opened in three student residences at the University of Pennsylvania in September 1988. This is a report of the study of those labs.

Background of the College House System

The College House system at the University of Pennsylvania consists of six houses and four living-learning programs whose main function is to promote faculty and student interaction and to underscore the value and potential of residential spaces as an extension of the educational space of the University. The oldest houses were founded in the early 1970's, and the program has developed steadily from that time. Currently, the program involves 22 resident faculty, 55 graduate fellows and about 1500 undergraduates. Recently the University has extended the College House model to create a series of first year houses, focusing on the needs and educational opportunities of first year students. Now more than half of all undergraduate residents on campus are living in a house program.

Each house and program has its own governing structure made up of committees composed of resident faculty, graduate and undergraduate students. Recently, the Council of College House Faculty Masters began a collaborative educational initiative to supplement the educational program of College House seminars, workshops and informal programs already in existence. The initiative seeks to encourage students to work together in groups to develop collective decision-making skills and to enhance and extend the abilities of individual house members. While the collaborative education program has a variety of features, the Council of Faculty Masters has underscored the important place of technology, and especially computers, in the emerging design of educational activities within the College Houses. The new Apple labs provide the starting point, therefore, not only of enhanced computer expertise for individual residents of the College Houses, but for developing support groups designed to foster writing and word processing skills, the acquisition of data from data banks using the University's central fiber optic computer network (PennNET), on-line library searches and other educational applications.[1]

Descriptions of Labs

The physical layout and use policies vary for each lab. DuBois College House has primarily Black students although it is designed for students of all ethnic backgrounds. Its lab contains 13 Macintosh SE's, one LaserWriter, and one ImageWriter. The lab is open from 4 p.m. to 4 a.m., and a monitor is on duty during those hours. The computers are arranged along three walls of a converted dorm room on the first floor. (See diagram.) Within the rough three-sided U created by the computers sits the lab monitor at a desk; the monitor's job is to sign out software and to help students who are unfamiliar with the computers. The LaserWriter sits between four SE's at the bottom of the U; the ImageWriter sits at the top left-hand side of the U. Seat cushions for the wooden chairs have been bought using house funds, along with framed prints to decorate the walls; a cassette player/radio sits on a shelf in the lab. The overall impression is that this is a comfortable lab.

Students of foreign languages live in Modern Language College House. Its lab is simpler. A small basement storage room was converted for use as the lab. It has five SE's arranged along one wall and a single ImageWriter. (See Diagram 1.) The MLCH LaserWriter is located across the hall from the lab. If students care to use the LaserWriter, they must sign-up for a specific time. Although on duty at all times, a monitor is not inside the lab. Instead, she sits at a desk in the hall immediately outside the lab. The lab is open from 4 p.m. to 2 a.m. No software is provided. A game room is located in the next room (approximately 20 feet from the lab) and beyond the game room is a TV room. The overall impression of this lab is that it is cramped and closed-in and fosters more of a gameroom rather than study hall atmosphere.

Stouffer is a house with no specific theme. Like DuBois, its lab is a converted dorm room but on the second floor and accessed by a maze of corridors. There are nine SE's, two ImageWriters, and one LaserWriter II. Three SE's are placed along one wall, with the remaining six ranging across two walls which join at about a 60 degree angle. The printers are placed against a remaining free wall, away from the computers. (See diagram.) Stouffer is alone in employing no lab monitors, though the two lab coordinators are active and often in the lab. The lab is always open. It is a bright and pleasant lab.

Profile of Student Computer Experience

An initial survey was conducted to learn about the computer access and expertise of the students in four of the College Houses. To some extent the use made of the new labs must be related to the students' need for computers and their ability to use them. 478 surveys were distributed to four College Houses during the last week in September, 1988 and 267 undergraduate replies were received in the first week of October. Although the major focus of this study was on the three houses with similar labs, a fourth house, Ware, was included in this survey to enlarge the profile base. Ware already had a mature lab with other types of micros but it received a few Macs from the Apple gift.

Return by Gender	
Female	55%
Male	45%

Return by Year	
Freshmen	18%
Sophomore	28%
Junior	22%
Senior	31%

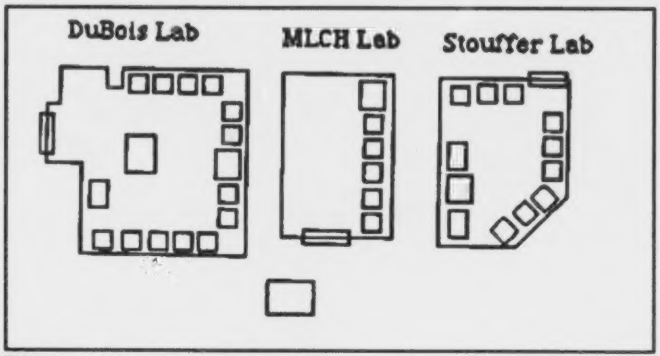


Diagram 1. Layout of the Labs

Access to computers

If students have their own computers in dorm rooms, will they make use of residential labs and if so why? Do students prefer to use other labs and if so why? Will students see residential labs as extensions of the academic life? Results of student use of residential computer labs are needed in order to plan for future computer implementation and for more comprehensive educational experiences for students.[2]

Equity is a major issue in the educational use of computers.[3] For that reason, data were analyzed for gender effects. Data were analyzed by year since differences are to be expected given the increasing availability of computers in schools and homes. Data were also analyzed by house.

Computer ownership. Results from this survey indicate that more than a quarter of the students living on campus (28%) own their own computers. Because students often share rooms, even more students have access to computers in their rooms (37%). About half of these computers are Macintosh, a quarter are IBM or compatible and a quarter represent a variety of other types.[4]

Differences in ownership. Freshmen and sophomores are more likely to have a computer in their rooms than are juniors and seniors ($p < .05$). Data indicate that personal computer ownership and dorm room access increase with each incoming class. More men (34%) reported owning a computer than did women (23%) ($p < .058$).

The presence of computers in dorm rooms varied significantly with the residences ($p < .01$). In Ware, 41% of the students reported owning a computer while in DuBois, the predominately Black house, it was 17%.

Computer Purchase Patterns. Most of the students who had computers had purchased them at home (61%). 21% purchased their computers at the University of Pennsylvania and 18% indicated that they purchased a computer elsewhere.

315

When computer was purchased	
Before coming to Penn	44%
Freshman year	23%
Sophomore year	9%
Junior year	16%
Senior year	7%

Previous Computer Experience

Home computers. More than half of the 267 students who responded to the September survey reported that their families owned computers. First and second year students were more likely to come from a home with a computer than third and fourth year students ($p < .05$).[5] There was no difference between men and women in this respect.

A significant difference ($p < .01$) was found between students in different residences in terms of home computers, however. 63% of the students in Ware reported a computer at home compared to 27% in DuBois, the predominately Black house.

In reporting their use of home computers, students in houses also differed significantly ($p < .05$). Although the majority of students reported that they used the computers in their homes, students in DuBois were much more likely to report never having used their home computer than were students in other houses. There was no significant difference in reports of time spent using home computers by women or men.[6]

Computer classes. Survey data indicate that each entering class arrives with greater computer experience than the last. 78% of the freshmen but only 59% of the seniors reported taking a high school computer course.[7]

No significant difference was reported between men and women in this respect nor was a difference found between students in the different residences.[8]

Additional experience. Two thirds of the students reported computer experience in addition to a high school course. Half of these reported that they had used computers related to a summer or college course and a third said that they had used computers on the job. Significantly more women than men reported taking courses and using computers in work ($p < .05$). Men, on the other hand, indicated greater informal computer experience. It seems that the nature of computer experiences may differ somewhat for men and women.

Students Perceptions of Computer Ability

80% of the students reported that they felt comfortable with computers. There was, however, a significant difference between men and women in this respect with more than twice as many women as men indicating lack of comfort ($p < .01$).

Current Computer Practices

Disk ownership gives some indication of computer use or expertise. 15% of the students indicated that they owned no computer disks while 25% indicated that they had ten or more disks. Most students (50%) reported owning between two and ten disks. Men were much more likely to own more than ten disks than were women ($p < .001$).[9]

Men were significantly more likely than women to report using a computer within the last day ($p < .01$) but no significant differences in the types of uses were determined. Men reported greater frequency of computer use with 24% of the males but 8% of the females reporting daily use.

Converging evidence supports the view that men and women and minorities bring quite different computer experiences to campus.

Word Processing

Because word processing is the overwhelming use that students make of computers, we asked specifically about it. Not only do a third of the students report that word processing is the only thing they have ever done on a computer, logs of use indicate that 83% of the activity in labs is word processing. In September, only 24 of the 273 students indicated that they had never used word processing.

How students learned to use word processing. Some students have been using word processing for many years, "Any serious paper I've ever written, I've word processed." while others are just now learning. "Extra time must be taken into consideration for computer problems." About half of the students who use word processing (49%) have been using it for more than two years while 34% have learned in the last year.

More than half of the students taught themselves how to use word processing (57%) while an equal number learned from courses or friends (18%). The rest indicated that they had learned in a job situation. Men and women show significantly different patterns in the way in which they have learned ($p < .01$). 61% of the men reported that they were self taught compared to 40% the women. Women are twice as likely to have taken a course to learn word processing as are men.[10]

Features of word processing most used. Word processing and related writing aids such as spell checkers encompass a great many features and we wondered which ones students use. Virtually everyone who responded reports using insert, delete, and block commands. Approximately three quarters of the people report that they use features such as page numbering, margin formatting and different fonts. Other features show more variation. 66% reported that they use a spell checker either often or sometimes but only 44% use a thesaurus. Students even report using some of the newer word processing features. 37% indicated that they had used an outliner but only 15% have as yet tried a style checker.

A significant gender difference was found in two of the word processing features. Women in our survey were less likely to report that they changed fonts and renamed or copied files.[11]

Effect of Word Processing on Writing. Over 50% of the students indicated that word processing had changed the way that they felt about writing. This is a strong statement that computers are having "an effect." The majority of students use word processing. 72% reported that they use it regularly with 36% who always use word processing for writing. Twice as many students reported that they often compose at the computer as those who never do. Computers and word processing are clearly beginning to change the way that students go about writing.[12]

In sum, the September survey showed that compared to men, women and minority students on campus were less likely to own a computer, used computers less frequently than men, expressed less self-confidence about computers and had fewer informal learning experiences with computers. This is consistent with national trends.

How the Residential Labs are Used

> *"I like the lab. I like the students in the lab. I think when we're in there there's this feeling that we're all doing work and that we're all being productive, that you don't get when you're sitting in your room, writing the same paragraph over and over." March 29, 1989.*

Methods of observation

Observation within the labs provides insights into social and computing practices that are common to all three labs. It also helps us to understand certain practices arising from conditions unique to the individual labs. In order to collect data in a reliable and standardized fashion, we developed and tested an in-lab observation form. Observations were recorded in five-minute segments, with the following information recorded: number of people in the lab; number of computers in use; number and type of programs in use; number of times printers were used; number of discussions between students and lab monitors; number of computer queries ("How do I move the tabs?"); number of social interactions (conversations with no connection to computers or school work); number of subject queries (questions or comments on school work). Nineteen and one half hours of visits were analyzed.

What Students Do in Labs

Categories of lab activities are listed below and the average number of occurrences for each category during a five minute period has been computed. These data, along with analysis of lab logs, allow us to describe what students do in labs.

Observations of Lab Activity
(Mean Number of Occurrences in 5 Minute Intervals)

D=DuBois (345 min)
M=Mod Lang (355 min)
S=Stouffer (470 min)
T=Total (1170 min)

	D	M	S	T
Lab Activities				
Print w LaserWriter	0.1	0.0	0.3	0.1
Print w ImageWriter	0.1	0.2	0.0	0.1
Word Processing	4.5	1.7	3.0	3.4
Graphics	0.0	0.1	0.1	0.1
Spread sheets	0.5	0.1	0.5	0.3
Games	0.0	0.0	0.10	0.0
Programming	0.1	0.1	0.1	0.1
Questions of Monitor	0.4	0.1	0.0	0.2
Computer Queries	0.1	0.3	0.6	0.4
Social Interactions	1.0	1.0	0.8	0.9
Subject Queries	0.6	0.4	0.7	0.6
By Whom				
Number of People in Lab	4.7	3.3	4.4	4.1
Computers in Use	5.0	3.0	3.7	3.9
Number of Women	3.4	1.4	2.1	2.3
Number of Men	1.2	1.8	2.2	1.8

Word processing is by far the most frequent activity and logs[13] confirm this (83% the first semester and 78% the second semester). Business applications (14% second semester) and graphics and programming represented the balance of use. Entertainment software was barely mentioned in logs.

Printing. There was not much printing observed in any lab. The time spent working on essays and projects far outstripped the time necessary to print them. Each lab provided unrestricted use of ImageWriters, but differences arose from strategies put in place to cope with the appeal of printing with the more expensive LaserWriters. DuBois charges 25 cents for each sheet printed using the LaserWriter; it does not charge for printing on the ImageWriter. Perhaps because of this, and also because of signs asking that the ImageWriter be used to print drafts, they keep the amount of printing with the LaserWriter less than that with the ImageWriter. MLCH has addressed LaserWriter expenses in a different manner, placing only an ImageWriter in the lab. Printing on the LaserWriter is done in a separate room, at assigned times, for 10 cents a sheet. Stouffer has made no such allowances in their lab. Use of printers is not monitored, and though, like DuBois, the Stouffer lab has signs designed to encourage students to use the ImageWriter for printing drafts, during the entire year we never observed the ImageWriter in use.

Use of software. In all three labs, word processing was the software most often observed in use. Average numbers of word processors in use at any one time rose in each lab from fall to spring. In DuBois spread sheets were second in popularity to word processing; some graphics programs and programming software were used; no games were observed. In MLCH spreadsheets and games were in evidence as well as graphics packages, but none was used extensively. The houses did not differ significantly in software use as reflected by observations or logs. The very small amount of time devoted to games is an indication that students think of the computer labs as places for serious work.

Who Uses the Labs? The total number of people using the lab at any one time is probably a factor of the number of students allowed to use the lab and the number of machines available. The policy of DuBois house is that only house members are allowed to use the lab, limiting the possible number of users to 100. With thirteen Macs and 1 AT&T, DuBois has the largest number of computers (ratio of .14). MLCH does not restrict use of the lab to house members, although it does not encourage outside use. There are 85 people in MLCH for five Macs, and five AT&T's (ratio .11). Neither does Stouffer restrict nor encourage use of the lab. There are 135 people in Stouffer for nine Macs, (ratio .06).

Our data indicate that MLCH made greatest use of its computers, using 60% of the available computers at any one time. Stouffer follows at 41% capacity and DuBois at 38%. These numbers reflect a change from first semester when Stouffer was clearly the lab most used by its residents. It makes sense that MLCH, the smallest lab, should be given heavy use. Lab use rose in DuBois and MLCH when second semester observations are compared with first semester figures; lab use in Stouffer dropped slightly.

Gender differences. More interesting is the ratio between female and male users of the labs. During the fall, DuBois women users outnumbered men approximately 3 to 1, in the spring the numbers remained the same. Given the high ratio of women to men in DuBois, this is not unexpected. In MLCH the ratio was almost even during the fall, and although more men used the lab in the spring, the ratio remained close to 1 to 1. The ratio of men to women using Stouffer during the fall was approximately 3 men to every 2 women. During the spring this ratio was more even, with the average for the year about 1 to 1. The results of our written surveys, which suggest that women are less likely than men to learn computer use on their own, when related to the fact that Stouffer was the one lab without constant supervision and aid, might help to explain the early low numbers of women users. The following chart lists potential users of the lab by gender.

Potential Lab Users by Gender

	women	men
DuBois	86%	14%
Mod Lang	55%	45%
Stouffer	50%	50%

Both men and women and both computer owners and non-owners use the labs. Although during the first semester we found that proportional to the male/female ratio of the residents, more men than women used the labs, by spring semester there was no difference. (Similar to U. Illinois finding, Palmer, 1988.)

Our survey data indicated that although more of the users of the labs did not own personal computers, heavy users (as defined by using the lab 10 or more times) were as likely to be people with computers in their rooms as not.

What Do Students Talk about in Labs? The figures for the number of computers in use at any one time are slightly below those for the number of people in the lab at the same time. This points to interaction that the lab fosters on a number of levels. Although students often left computers turned-on and unattended when they left the lab for short study breaks (these computers were recorded as in use), the number of people working together on computers, either demonstrating or learning software or working on projects together, was great enough to bring the number of computers in use below the number of people.

Many conversations began with a question about how something worked, although we observed slightly less of this during the spring. These queries were directed to the monitors if one were present else to another person in the lab. Conversations frequently began when someone glanced casually at a neighbor's monitor and asked "What are you working on?" This would often lead to discussions about the work at hand. Talking regularly took place while students were waiting at the printer. The owner of the paper would ask others what they thought of it.

How Long Do Students Stay in the Labs? Based on logs, the average length of time that students used the lab came to approximately 2 hours. This average, however is balanced between the many users who came in for half hour periods, often to use the laser printers, and those who stay for long stretches of 6 or 7 hours of work.

When Do Students Use the Labs? When we asked students what would be a good time to visit to see people working in the computer labs, we were universally told to come late at night during the week. Logs confirmed that the greatest use was from about 9 pm to midnight (11 P.M. was the busiest hour), but if students had access to the labs, they could be found there at all times of the day and night. Friday evening through Sunday afternoon was the least active time in the labs.[14]

The hours that computer labs are open seems to be the most important lab issue for students. Responses in our surveys for suggestions for improving the labs again and again mentioned increasing lab hours in those houses where use was restricted. Responses to questions asking how the lab had helped also solicited a majority of responses referring to access. As one person stated,

> *"The most important feature is that it is available 24 hours a day."*

Student Views of Residential Labs

> *"You get to know what other people in the lab are working on and you might find common interests and problems."*

> *"It helps to have people to ask questions and to get to know people: it relieves stress."*

Interviews with students and surveys in the three houses in December and April provided information about what the students thought about their labs.

94% of the students responded that the labs were good places to work and 80% reported that they had asked for technical help while working there. Three quarters replied that they had learned more about word processing features, while over half of the students reported that they had discussed the content of their work with other people in the lab. It would appear from these results that the labs were more than just places to work, that they were also places where students received technical support, learned more about computer programs and often received feedback on the content of their work. Three quarters of the students indicated that the availability of a computer lab would influence their decision in picking a residence.

Differences in Labs

Lab organization may be related to how helpful people in the labs are perceived to be. For example, 100% of the replies from DuBois indicated that people in the lab were helpful. DuBois had a trained monitor on duty at all times. 94% of replies from Stouffer suggested that people in the lab were helpful. In Stouffer informal experts who had their own computers were often in the lab. In Modern Language house only 65% of the responses indicated that people in the lab were helpful. Because of the small size of the lab in this house, monitors were located outside and there were fewer people to ask for help. In response to a question about whether they would prefer a computer in their room to a computer lab, 47% of students in DuBois preferred the lab, 40% in Stouffer but only 28% in Modern Language House.

Women Increase Lab Use During Spring Semester

As noted in our observations, the number of women using the labs increased during the spring semester. (University of Illinois noted similar change. Palmer, 1988). We suspect that residential labs are particularly well suited to provide the types of informal computer learning experiences that our first survey (and national data) indicate have been lacking for women and minorities. Significantly more women reported that they asked for help while working in the lab than did men ($p < .05$). A significantly higher percentage of women also reported that they used the lab because they didn't have their own computer ($p < .05$). More women reported that they discussed the content of their work when in the lab than did men.

Students Who Own Computers Use Labs Too

A surprising result was the fact that some students who had computers in their rooms reported using the residential labs frequently. Of students replying to the survey who owned computers, 48% replied that they had used the computer lab more than ten times during the year. This compares with 51% of students without their own computers who reported using the lab more than ten times. We know that the laser printers were a draw, but the students' comments lead us to believe that the friendly and supportive environment of the house labs make them attractive places to work.

> *"My knowledge of using the computer has stemmed from this house's lab. Papers are not really as tedious and they eliminated my fear of computers."*

In sum, student opinions of the labs were overwhelmingly positive. The lab in DuBois which always had a monitor and to which students brought pillows and music was perceived as the most helpful of the labs. Women's use of the labs increased during the spring semester and women were significantly more likely to report that they had sought help in the lab. Students who owned their own computers also used the labs. Student perceptions seem to indicate that the computer labs had indeed become areas for informal learning and extensions of university learning space.

Conclusion

The three computer labs that were observed during this study were set up in very different ways. Given the differences, many of which were attractive for some reasons but unattractive for others, we would suggest that people considering implementing residential labs give careful consideration to the following areas:

1. Choice of room: The two converted dorm rooms were more successful and pleasant than the converted basement storage room.
2. Hours of access: Students overwhelmingly wanted 24 hour access.
3. Monitoring: The most efficient and most pleasant lab was DuBois to which there was no admittance without a monitor. A monitor was assigned for regularly scheduled house. During other times,

students could use the lab if they could locate a monitor. This combination of monitoring and self-monitoring worked well.

4. Software: For labs to reach their potential and support communities of learners, software and other new technologies such as CD ROM and videodiscs should be available.

We are rapidly approaching the time when computer ownership on campus is as widespread as pencil ownership. If we view residential labs as facilities that make computers available to students who do not yet own them, then they are indeed a temporary phenomenon on campus. If we view these labs as extensions of university academic space, the direction that the results of this study have led us, the continued existence of these labs is important. We came to view these labs as places which encourage students to work within a community of scholars or writers, places which discourage to some extent work as solitary scholars or writers. Slipping almost seamlessly from social, or house oriented, conversation to academic subject matter and back again was the characteristic flow of discussion in residential labs. An informal supportive workshop atmosphere developed in each of the labs.

The amount of informal learning about hardware and software that we observed is another reason to consider the importance of residential labs. It is unlikely that the pace of either hardware or software development will slow, and the need to keep learning appears to be endless. The way by which the majority of students reported that they had learned word processing, for example, attests to the informal nature of much computer/software learning. Residential labs seem particularly well suited to serve this role, and they also seem likely to help narrow the gap in computer experience with which students arrive on campus.

A student expressed it well:

> *"Well, as a house we study together. It is very social. We have the radio going and we talk about the song, whatever. So as a house it brings us closer together because we study together. Before the lab we didn't study so much together as far as if you had something to type. If you are an engineer you use the computer and now if you're in English you use the computer, so now we study in the lab together. Last year we all studied in the multi-purpose room, which people still do, but more people study in the computer room. As a house we have become closer. Also, people who were completely computer illiterate I've seen become very comfortable writing on Macs. And I like to think that is helping people's grades. The house GPA has gone up. I'm not saying that that's totally due to the computer room. But I don't think it is hurting."*

Notes

[1] See University Publication, "Penn, Residential Living, Undergraduate Options" for a full description of each house.

[2] Reported by W. Ziller (1987). The National Institute of Education and Carnegie Foundation reports: A residence life perspective. *Journal of College and University Student Housing*, 17(2).

[3] Becker, H.J. & Sterling, C. (1987). Equity in school computer use: National data and neglected considerations. *Journal of Educational Computing Research*, 3(3), 289-311.

[4] These figures are consistent with data collected by the Computing Resource Center, U of P. Personal communication, Jeffrey Seaman, Dir. C.R.C., November 1988.

[5] This result is consistent with home computer availability described in Ancarrow. J. (1987). Use of computers in home study. Center for Educational Statistics, U.S. Department of Education.

[6] This is interesting in light of other studies showing that boys are more likely to be the users of home computers. Ancarrow. J. (1987). Use of computers in home study. Center for Educational Statistics, U.S. Department of Education.

[7] This result is consistent with data reported from Syracuse University. 73% of Freshman reported taking a high school computer course (Kent, P., September 1988). The evolution of a computing assistance program for faculty Academic Computing . Also consistent with data reported in *Power On! New Tools for Teaching and Learning (1988)*. Congress of the United States Office of Technology Assessment. In the near future we can expect that 100% of students will arrive on college campuses with computer experience.

[8] Our questions did not address the nature of the high school courses. Although nation-wide there are gender differences associated with computer courses (girls take data entry courses for example), according to Becker, no differences are reflected in high-ability programming classes of the sort that Penn students could be expected to have taken. Henry Jay Becker (August 1986). Instructional uses of school computers: reports from the 1985 national survey. Johns Hopkins University.

[9] From this we might infer that men are likely to be more involved in computer use. Hess, R. & Miura, I. (1985). Gender differences in enrollment in computer camps and classes. *Sex Roles*, 13 (3/4). 192-203.

[10] This result seems consistent with the earlier question indicating that women report less self-confidence in computer use.

[11] This result is puzzling. Perhaps file management is related to disk ownership. Our data indicate a gender difference in this. The changing fonts result is even

more puzzling. It is consistent with data collected with a class of elementary school children in another research project in which analysis of computer dialogues showed that boys spend more time talking about and changing fonts than girls.

[12] We suspect that there is a relationship between access, length of time of word processing use and typing skill with composing at the computer. Students who have achieved success composing with paper and pencil would not have reason to quickly drop this method.

[13] Although DuBois House regularly kept logs of lab use, it was only at our insistence that Stouffer and Modern Language kept any records of lab activity. We asked that logs be kept for one week each semester but compliance was voluntary and we suspect that some students were concerned that we were attempting to monitor individual activity even though we made every attempt to dispel that worry.

[14]. Our results are consistent with residential lab use at the University of Illinois as reported by C. Palmer, "Residence hall computer utilization report: A summary of major findings," May 1988).

Bibliography

Ancarrow, J. (1987). Use of computers in home study. Center for Educational Statistics, U.S. Department of Education, Washington, D.C..

Becker, H.J. & Sterling, C. (1987). Equity in school computer use: National data and neglected considerations. *Journal of Educational Computing Research*, 3(3), 289-311.

Becker, H. J. (August 1986). Instructional uses of school computers: reports from the 1985 national survey. Johns Hopkins University, Baltimore, MD.

Daiute, C. (1985). *Writing and Computers*. Addison Wesley.

DeLoughry, T. J. (1988). Once they asked: How many library books? Now it's: are computers available at 3 a.m.? *The Chronicle of Higher Education*, September 14.

Kent, P. (1988). The evolution of a computing assistance program for faculty. *Academic Computing*, September, 28-31.

Lutz, J. (1987). A study of professional and experienced writers revising and editing at the computer with pen and paper. *Research in the Teaching of English*, 21(4), 398-420.

Miura, I. (1985). Gender differences in enrollment in computer camps and classes. *Sex Roles*, 13 (3/4), 192-203.

Palmer, C. (May 1988). Residence hall computer utilization report: A summary of major findings. University of Illinois, Champagne/Urbana, Illinois.

Power On! New Tools for Teaching and Learning (1988). Congress of the United States Office of Technology Assessment, Washington, D.C.

Treisman, Uri (1989). Talk given at the University of Pennsylvania (March).

Ziller, W. (1987). The National Institute of Education and Carnegie Foundation reports: A residence life perspective. *Journal of College and University Student Housing*, 17,2, 8-12.

Improving the Writing Ability Of Middle Level Students Through a Role-Taking Experience Using Computers

Jane D. Steelman
North Carolina State University
Dept. of Curriculum and Instruction
Box 7801, 402 Poe Hall
Raleigh, NC 27695-7801
(919) 737-3221

Abstract

The purpose of this intervention study was to implement and evaluate an instructional program in writing with computers for middle level students. The novel approach to the teaching of the writing process was in its presentation through a role-taking experience or project approach to promote student development. The technology of computers was utilized in order to facilitate the revision (reflection) strategies so important to the writing process and cognitive development. The curriculum intervention was employed for approximately one school year (28 weeks).

A pretest-posttest design was chosen for the present study employing two experimental groups and one control group. The two experimental groups were compared to test for the effects of the computer alone. The experimental group not using computers was compared to the control group to test for the effects of the role-taking writing program alone.

Measures to determine writing ability, writing apprehension, and writing quantity were employed. A pretest and posttest one-way analysis of variance (ANOVA) was applied to discover significant differences between groups (alpha level of .05) on each measure. Fisher's LSD procedure was applied to determine which groups differed significantly. Student written remarks were evaluated in order to determine how the students felt about the interventions in which they were involved.

The results are discussed with respect to the effect of the writing process approach and the potential of computers and word processing programs for providing an enriching environment for the teaching of writing in the middle level classroom. The effectiveness of the intervention is discussed with respect to differences in writing quantity, quality and writing apprehension.

Introduction

Even though the writing process has been advocated for many years, a more traditional approach to teaching writing remains prevalent in the classroom. The emphasis upon writing as a process rather than a product has been advocated for many years. Much research has been published which shows the positive effects of the writing process approach; however, composition instruction, especially in the area of exposition, is still underdeveloped (Newton, 1985). Writing which focuses on the process rather than the product and makes use of the new writing models and the new technology of computers may improve writing achievement (Withey, 1983). With the development of improved word processing programs for children, the use of computers in the writing process has become more prevalent.

The research accomplished up to this time indicates the effectiveness of the computer in the writing process but few of these have incorporated an intensive training program in how to use the word processing tool's full capabilities prior to any actual intervention. Many programs have been developed which try to relate the writing process to a developmental framework. The implementation of these programs usually lack an experimental design and tests for the results of the program. The programs are usually implemented by the researcher/designer of the intervention and try to change only the method of delivery of the same content without employing an effective role-taking experience. The majority of the programs developed to improve writing along developmental lines have been utilized with college-aged individuals, usually in freshmen composition classes (Hays, et al.,1983). Even though the qualitative analysis and anecdotal comments from students and teachers have been positive, a need for more controlled research employing control groups, various objective tests, and younger students (middle level) is indicated.

The study done by Carnew and Clark (1985) did implement a computer intervention for writing, as well as other subjects, with adolescent students with significant positive results favoring the experimental groups. The study was restricted to a specific cultural group and was based upon cognitive-developmental theories different from those in the present study.

Method

Subjects

The students involved in this study were chosen from a relatively small rural school system in a southeastern state. The community from which the students were chosen is made up of primarily lower socio-economic status individuals. The population of the community consists of a majority of minority race members. The sample was comprised of 75 sixth grade students from one middle school in the school system. The school contained grades four through six with three and one-half classes at the sixth grade level. Each teacher taught two or three subjects within a block of time for all students in the sixth grade; therefore, one teacher taught all students language arts. The students were assigned to one of three classrooms randomly by gender and race.

Procedure

The intervention took place over approximately 28 weeks from September to March within two of the intact sixth-grade classrooms. The experimental groups were involved in the newspaper writing program two days per week, two hours each day for a total of four hours per week. A control group receiving no intervention was also be employed. The language arts classroom teacher instructed the three groups involved in the study. The researcher acted as a teacher assistant and instructed the students in the use of the computer.

Experimental group using computers:

Phase 1: The students were instructed on how to use the computer appropriately. They were asked to do certain activities which deal with the operation of the machinery and the handling of diskettes. On successful completion of this phase each student received a "computer operator's license."

Phase 2: The students were instructed in keyboarding skills. The students were instructed on where their fingers should be placed on the keyboard when typing in order to improve efficiency when writing with the word processing program. A program produced by Broderbund, *Type!*, was employed to give each student individualized practice in keyboarding (Eicholz, 1987). The program was available during class time as well as before and after class.

Phase 3: A two-week intensive program on how to use *Bank Street Writer*, (Bank Street College of Education, 1986), a word processing program designed primarily for students in the middle grades, and revision strategies were employed prior to the implementation of the newspaper writing curriculum using computers. The revision strategies were those included in the *Bank Street Writer Activity Files*, Volumes 1 and 2 (1984).

Phase 4: The newspaper writing curriculum utilizing the computer for two days per week, two hours per day was implemented. During these sessions the students worked on many different aspects of writing a school newspaper,such as gathering necessary information about subjects to be included in the paper, writing drafts, peer editing, writing final copies, producing visuals, and publication.

The students worked together with each other and the teacher to decide upon some some relevant topics of interest to them and to their schoolmates to be included in the newspaper. Brainstorming techniques were used to develop a list of ideas of interest to the students. These ideas were then be put into a priority list and used as a guideline in developing articles for the newspaper.

The students met in pairs or small peer response groups to critique and discuss each others' writing. The students were guided by a structured outline for critiquing each others' work until they demonstrated that they could comment effectively without being destructive to each other. The students eventually were able to critique each others work without the structured guidance.

The newspaper was published once every three or four weeks for all members of the student body and their parents. In this way students took on the role of reporters, editors and publishers. They were challenged by the fact that their work was published for all to see. The definition of this audience challenged and motivated the students to take seriously the roles which they assumed. The newspaper writing experience was a collaborative effort within the classroom, emphasizing a team effort and allowing the students to work together to create a product important to them. The production of a newspaper enabled the students, parents, and administrators to view "the fruits of our labor." It is through publication that the student really emulates the role of a writer and is able to put him/herself in the writer's place.

Experimental group not using computers:

The newspaper writing curriculum as described above, using only paper and pencil techniques was implemented for two days per week, two hours per day. A two-week program on how to cut and paste on paper using identical revision activities were presented instead of how to use the computer/word processor prior to the implementation of the role-taking writing experience. Both groups contributed to a single school newspaper rather than two smaller publications.

Control group:

The students in the control group met in writing class for two days per week, two hours per day. They used prewriting motivators developed by the instructor.

The control group was presented with traditional writing instruction previously used in the regular language arts curriculum. The regular language arts curriculum consisted of a reliance on textbooks adopted by the state for language skills.

Many teachers, including the instructor involved in this research, have developed some familiarity with the writing process approach to the teaching of composition. Many classroom teachers tend to focus on certain stages of the writing process without seeing their efforts come to a logical conclusion due to the perceived lack of time in the classroom. It is the contention of this researcher that the use of only parts of the writing process result in minimal writing achievement. It is the process of working through *all* of the stages of the writing process which facilitates development in writing. One must have all of the elements of challenge, support, continuity, balance, and role-taking in order for a developmental model to be effective. A more intense implementation of the writing process and the involvement in role-taking can produce greater improvements in writing ability.

Instruments

Attitude toward writing was determined using Daly's Writing Apprehension Test (Daly & Miller, 1975). An average number of words per writing sample were compared for each group to determine significant differences in the amount of writing produced by those students in each group. For all three groups of students, the number of words per writing sample were determined in order to compare the quantity of writing done by each student from the pretest to the posttest. A total number of words per writing sample (pretest and posttest) were compared for each group to determine significant differences in the amount of writing produced by those students in each treatment and control group. A gain score was also evaluated to determine the effects of the writing program on writing quantity.

Point Score	Characteristics
8	Has an innovative theme Concrete details used effectively Fluent in words and ideas Varied sentence structure with complex sentences Effective closing statement Clear mechanics with few or no errors
7	Has a theme Concrete details used effectively Fairly fluent in words and ideas Varied sentence structure Satisfactory closing statement Generally clear mechanics
6	Has a central idea Specific facts, details, or reasons Consistent development Less insightful, imaginative, concrete, or developed than a *7* Generally clear mechanics, errors do not interfere with overall effectiveness
5	Has several clear ideas Relevant and specific details Evidence of fluency, but not of unified development May be overly general or trite May have simpler sentence structure and/or vocabulary than a *6* Mechanical errors do not affect readability
4	Has at least one idea, few, if any, supporting details Less fluent, developed or detailed than a *5* Sentences, vocabulary, and thought are more simplistic than a *5* Mechanical errors do not affect readability
3	No clear theme Has a sense of order, but order may be only that of plot summary Fluency and thought are minimal (less fluent, well-developed, or detailed than a *4*) Has at least one relevant idea May have many mechanical errors but paper is readable
2	No theme and, of course, no support for theme Little sense of organization Simplistic or vague language May or may not have one relevant idea May be unreadable due to spelling, handwriting, or other mechanical problems
1	No theme and no support for theme No sense of organization No relevant ideas Simplistic language (less complex than a 2) Difficult to follow or unreadable
0	Unscorable, Unreadable Nonsense words or sentences

Table 1. Scale for Scoring Essays of Experiment and Control Groups

Each writing sample was scored using an eight-point holistic scoring guide or rubric. When two samples were combined, there was a possible score range of 0 to 16. This researcher employed the analytic scale procedure for holistic evaluation of writing. In the analytic scale procedure the writing is scored by the way it fits within a range of papers for the assignment using a general guide of what should be present in the writing to warrant a particular score. This holistic procedure is particularly appropriate for program evaluation or research on methods of teaching writing (Cooper & Odell, 1977).

Two writing samples taken on different days were obtained from all students in each group. Scores on writing samples were evaluated using an eight-point holistic scoring guide or rubric (see Table 1). The rubric was developed by the researcher based on previous rubrics used by other researchers at this university (Pritchard, 1987; Wester, 1985). Each writing sample was given a combined score determined by three raters to produce a seventeen point scale from 0 to 16.

In using the researcher-developed rubric, three raters were engaged to evaluate responses to the writing prompts. The two ratings which did not differ by more than one point were used to determine the score for each sample. This procedure was used to improve inter-rater reliability and insure that consistent results were obtained.

Data Analysis

A one-way analysis of variance (ANOVA) was applied to discover significant differences between groups on writing ability, writing quantity, and writing apprehension. An alpha level of .05 was used to determine if significance existed. The LSD procedure was used to determine which groups differed significantly.

Results

Pretest analysis of variance procedures determined that there were no significant differences on any of the reported measures between the groups prior to the intervention. Posttest analysis of variance procedures supported the hypotheses that the use of the writing process approach is superior to traditional methods of teaching writing and that the use of the computer enhances the process. There was a significant difference in quality of writing produced by children in the experimental groups compared with the control group. Table 2 represents mean scores for each experimental group and the control group.

The ANOVA performed using writing ability as the independent variable yielded a significant main effect for group, $F(2,64) = 3.5631$, $p < .03$. The LSD indicated that the group using computers differed significantly from the control group and the group writing the newspaper without using the computer also differed significantly from the control group.

Posttest

Group	n	mean	std dev
Experimental 1[1]	22	7.886	3.306
Experimental 2[2]	25	7.280	3.536
Control	20	5.200	3.350

Note: Scores on the two combined writing samples range from 0 to 16, with a higher score indicating greater ability.

[1] Experimental 1 = group using computers to write the school newspaper

[2] Experimental 2 = group writing the school newspaper without the computer

Table 2. Writing Ability

Students in the experimental groups wrote significantly more than those in the control group. Table 3 displays the mean writing quantity for pretest, posttest and gain scores.

The ANOVA performed on the posttest writing quantity scores yielded significant differences between the experimental groups and the control group, $F(2,65) = 3.25$, $p < .04$. When using the amount of writing done from pretest to posttest, i.e., gain score, the ANOVA performed indicated significant differences between each group, with the group using the computer to write the newspaper scoring significantly higher than the other two groups, $F(2,59) = 14.71$, $p < .0001$.

No significant differences were found for writing apprehension. The mean posttest scores are shown in Table 4.

Discussion

The writing ability of students is important to many tasks which they undertake in and out of the school environment. The present study indicates that the use of the writing process is effective in improving the writing ability and the amount of writing done by students in the middle grades. The computer can also play an important part in enhancing the teaching of writing to students in the middle grades. The experience of creating a school newspaper added a dimension to the writing curriculum very important to middle level students. They could see that what they were creating was meaningful and read by many other people which translates to viewing writing as a worthwhile effort.

It was predicted that the students in the experimental groups would express less apprehension toward writing

Posttest

Group	n	mean	std dev
Experimental 1	22	295.0000	128.3852
Experimental 2	25	258.8800	141.1785
Control	21	197.4286	104.9317

Gain Score

Group	n	mean	std dev
Experimental 1	19	89.9474	65.0303
Experimental 2	23	29.0000	68.3560
Control	20	-57.9500	115.8677

Table 3. Writing Quantity

than the control group. This prediction was not confirmed; the control group began with a lower anxiety level and the means remained approximately the same from pretest to posttest for all groups. This may be due to the rather low anxiety expressed by all students prior to the intervention. There is also a certain amount of anxiety produced by using the computer and by writing a school newspaper.

Both experimental groups differed significantly from the control group with respect to writing quality. Mean scores for group 1 (with computers) were greater than group 2 (without computers) and group 2 (without computers) mean scores were greater than mean scores of the control group. These differences indicate that the computer alone may not make a significant difference but when used in conjunction with the writing process it can become a powerful tool.

The computer lab in the school had only 7 computers to be shared by 30 students at a time. It was necessary to rotate the students to the computers individually or to work in pairs on articles for the newspaper. The results of the study are encouraging since the researcher believes that an environment in which more computers are used could certainly produce greater gains in writing achievement. Regardless of the limitated resources, the computer using group tended to write longer papers than the other two groups and this increase in amount of writing could lead to increased quality over time.

Implications

A major goal of education should be to produce students who can think creatively and express their ideas effectively to others. Student writing must be encouraged throughout the curriculum, but especially at the middle level since these pre-adolescents have a great need and desire to express themselves. The use of a major project which gives the students experiences beyond the school environment can be very important to providing middle level students with "real" situations which they may see as worthwhile. The participation in projects the students see as worthwhile may increase their motivation and involvement.

The experimental group using computers had higher mean scores than the other two groups and differed significantly from the control group even though there were limited computer resources. This indicates a need to provide students with more computer resources and more time to use computers in word processing across the curriculum. Further studies should be done in school environments which have greater computer resources. It is important that future studies implement a plan of teaching the word processor, revisions strategies, and keyboarding skills prior to the implementation of the writing project. The developmental nature of the writing process indicates a need for any researcher undertaking a project to improve writing ability to devote at least one school year to the effort. Prior studies dealing with the writing process and computers may have been too brief to demonstrate significant differences. This researcher found that the preliminary activities of teaching the word processor, keyboarding, and presenting revision strategies took as much time as the total time devoted to many of the previous studies. This researcher believes that the preliminary activities were necessary to the study in order to see if the program implemented really had a positive effect rather than putting the

Posttest

Group	n	mean	std dev
Experimental 1	24	94.6667	18.5675
Experimental 2	26	93.2308	15.7132
Control	24	98.7083	11.6376

Note: Scores on Daly's Writing Apprehension Test range from 26 to 130, with a higher score indicating a lower anxiety.

Table 4. Writing Apprehension Scores

students in a more stressful environment of trying to learn the new technology as well as to produce a product. The program could not be effectively evaluated when one is really evaluating the students ability to cope with an enormous amount of challenge.

References

Bank Street College of Education, Smith, F. E., & Intentional Educations, Inc. (1986). *Bank Street Writer* [computer program]. New York, NY: Scholastic Software.

Bank Street College of Education, Smith, F. E., & Intentional Educations, Inc. (1984). *Bank Street Writer Activity Files, Volumes 1 & 2* [computer program]. New York, NY: Scholastic Software.

Carnew, F. I., & Clark, W. B. (1985). Cognitive Education and Native Adolescents: A Pilot Study. Calgary University (Alberta). Institute for Computer Assisted Learning. (ERIC Document Reproduction Service No. 268 989).

Cooper, C. R., & Odell, L. (1977). *Evaluating Writing: Describing, Measuring, Judging.* Urbana, Ill. NCTE.

Daly, J. A., & Miller, M. D. (1975). The Empirical Development of an Instrument to Measure Writing Apprehension. *Research in the Teaching of English, 9*, 242-249.

Eicholz, B. (1987). *Type!* [computer program]. San Rafael, CA: Broderbund Software, Inc.

Hays, Janice N., Roth, Phyllis A., Ramsey, Jon R., Foulke, Robert D. (1983). *The Writer's Mind: Writing As A Mode Of Thinking*. Urbana, Ill: NCTE

Newton, S. S. (1985). Using the Word Processor in Composition. (ERIC Document Reproduction Service No. 262 943)

Pritchard, R. J. (1987). Effects on Student Writing of Teacher Training in the National Writing Project Model. *Written Communication.* 4(1), 51-67.

Wester, J. (1985). *Cognitive-Developmental Theories in English Composition: Program Description and Evaluation*. Unpublished doctoral dissertation, North Carolina State University, Raleigh, N.C.

Withey, M. M. (1983). The Computer and Writing. *English Journal*, 72(11), 24-31.

Learning Japanese Literature Through a "Hyperfilm"

Hiroo Saga and Yasuki Hamano
National Institute of Multimedia Education

Masaaki Hagino
Pioneer LDC, Inc.

Abstract

This presentation demonstrated a HyperCard-based interactive video for learning Japanese literature. In our product, named "Bunkyo Museum of Literature," multiple contents and symbol systems were integrated to give learners access to a rich environment of a literary world. We used an existing 16mm film as a main material for this development because we wanted to give a rich and dense context to the whole environment. The film, "Great Authors of Bunkyo," describes the life or Mori Ogai, who lived in the Bunkyo ward of Tokyo and was one of the great authors in the Meiji era and in his circle of acquaintance with other authors.

The film was transformed into a videodisc and linked to the stacks of information on HyperCard. The developed software allows full control of the videodisc as well as flexible interaction with the following 10 stacks; 1) Bunkyo Museum of Literature, the main stack containing a film theater; 2) Film Scenario; 3) Authors of Bunkyo; 4) Works of the Authors; 5) Chronologies of the Authors; 6) Ogai's Album; 7) Maps of the Authors; 8) Reference Articles; 9) Reference Pictures; and 10) Guide to the Museum.

Two types of formative evaluation are now conducted using college and high school students. One focuses on individual paths of the learners as related to their personal traits. The other examines aspects of classroom instructions using this medium. Initial findings of these studies will also be discussed.

Sights, Sounds, Science, Stacks: Hyperstudio and Interactive Video

E. Byron Rogers and Lynn M. Mason
Lubbock Christian University

Abstract

HyperStudio by Roger Wagner Publishing is the hypermedia authoring system for the Apple IIgs computer. With this software, teachers may create an endless variety of applications incorporating text, color graphics and animation, and digitized sound. With the addition of a laser disc player, interactive video software may also be produced. The *HyperStudio* system is easy to use and is appropriate for use in virtually any discipline.

We have developed a *HyperStudio* stack for use with the Periodic Table Videodisc by the Journal of Chemical Education. This stack allows the user to view the broad properties of the periodic table of elements on three basic screens, with accompanying on-screen text files. The user may also look at individual element information, including text as well as still photos of the element, real-time reactions, and still photos of uses of the element (from the videodisc).

This stack was designed for use by high school and lower level college instruction, but would also be suitable for use in a middle school/junior high science curriculum. All of the text files used by the stack are editable, so that the information presented may be tailored for the specific student audience.

The Effects of Progressive Levels of Interactivity in an Interactive Video Program on Learning

Margaret Lynn Bailey and Jackson Byars
Kansas State University

Abstract

Results of a research project that investigated whether student pairs assigned to one of three levels of interactivity (none, low, or high) in an interactive videodisc lesson would have significant differences in their verbal interaction and overall learning of the lesson content will be discussed. The three interactivity variations were created using the Learning Systems/1 authoring language for the IBM InfoWindow System and were based on the physics videodisc *The Puzzle of the Tacoma Narrows Bridge Collapse*. Data was collected and analyses made on students' retention and transfer of the lesson material, attitudes toward the learning environment, and frequency and type of verbal interactions. Preliminary results show no significant difference in retention or transfer between the three levels of interactivity.

Development and Use of Interactive Video CAI to Prepare Community College Students for Minimal-Competency Examinations in Mathematics

Zan Tamar Bailey
Caber Systems

Cynthia A. Elliott
Miami-Dade Community College

Abstract

Interactive videodisc is now widely recognized as an innovative instructional technique for directing and assessing student learning. Miami-Dade Community College is using this technology for individualized tutorial instruction for class preparation and for preparation for the CLAST, a Florida minimal-competency examination required for all degree-seeking students.

The purpose of this project presentation is two-fold: 1) to describe *Interactive Math*, a level III interactive videodisc tutorial developed and presently in use at Miami-Dade Community College, and 2) to discuss the results of research conducted to determine the effectiveness of *Interactive Math* when used to help Miami-Dade students prepare for the CLAST.

Results of the research to determine the effectiveness of the *Interactive Math* series have shown a significant difference in the CLAST results (higher mean test scores) for the group using *Interactive Math* in addition to traditional classroom instruction when compared to a control group who did not use *Interactive Math*.

This presentation will include a brief discussion of interactive videodisc (IVD) hardware and the educational capabilities/uses of IVD, a demonstration of *Interactive Math*, and a discussion of the results of research conducted to determine the effectiveness of the *Interactive Math* series.

Activities to Motivate Women to Use Technology in the Humanities

Vanessa Evans Huse
Lon Morris College

Abstract

Throughout the last decade, as our society progressed into the information age, there has been a decline in liberal arts education with an increase in technological fields. Historically, women have avoided scientific fields and have been attracted to various fields of the humanities. Few women have crossed the barrier into our new technological world, with only 15% of all bachelors degree candidates entering into the workplace with degrees in mathematics and technology fields. However, in today's world where 98% of the jobs in America require the use of a computer, no field of study is exempt from the use of technology.

The goal of the presentation is to report on an extensive search of the various studies involving motivating women in mathematics and science. After all factors were identified, a proposal was constructed to determine which of these factors can be used to encourage women to use technology in the humanities. The outcome of this project would enable educators in secondary schools and colleges to introduce motivating activities that will stimulate women to learn about technology, and to create environments where women would feel comfortable incorporating technology in their chosen field of study.

Computer Learning in Adults: Using Kolb's Adult Learning Styles Inventory to Assess Adult Computer Learning

Jill H. Ellsworth
Southwest Texas State University

Abstract

This research project tested the hypothesis that computer learning in adults at the college level is influenced by the fit between the learning style of the adult students and the teaching style (methodology). The outcome intention is to adjust teaching styles as opposed to trying to match teaching and learning styles in course sections.

The project used Kolb's Learning Styles Inventory (LSI) for assessment. The LSI identifies the adult's learning in four areas: concrete experience, reflective observation, abstract conceptualization, and active experimentation.

Two teaching styles were utilized: didactic and interactive. Didactic teaching involved cognitively focussed lecture/demonstrations followed by self-paced labs for students assigned as homework. Actual hands-on computer work was unsupervised. Interactive teaching involved experientially presented lecture/explanations with in-class computer practice: Students were taught at the computer. Grade/style correlations were used to assess success.

The 205 adult students were aged 19 to 72, with the largest number falling between 28 and 42; 99 were male and 106 were female. These students were enrolled in the same course with the same instructor; the study was conducted across four semesters. The same text, software, tutorial and supplemental materials were used in all classes.

Analysis indicates that the interactive/experiential method was superior with all four learning styles, but particularly successful with two of Kolb's types: accommodators and divergers.

Ethical Insight of Undergraduates: The Influence of Family, Friends, Religion, and Previous Education on Freshman Understanding of Ethics

Janet M. Cook
Illinois State University

Abstract

Recent crimes and viruses have focused the attention of computing professionals on the way that students acquire standards of "right conduct." Before we plan to enhance our teaching of professional standards, we should know what standards students have when they enter college.

A study conducted at Illinois State University in Fall, 1989 polled over 600 students enrolled in 100-level courses to assess their own perceptions of their ethical background and opinions. The results broke many stereotypes about the influences of age and gender, family and religion, personal and professional ethical standards.

Extending Preservice Teacher Preparation Programs with Distance Education Technologies

Judi Harris, Chair
University of Virginia

Abstract

University teacher preparation programs are beginning to infuse distance education technologies into their preservice curricula, breaking the artificial barriers created by classroom walls. Computer mediated telecommunications networks, for example, are used to store and provide easy access to course materials, present course content, build support structures and bridges between schools of education and local public schools, and facilitate development of curricula for community schools.

A panel of experienced educational telecommunications facilitators with international institutional affiliations will discuss their experiences with incorporating computer mediated communications into preservice programs in education during this session. They are:

Rhys Gwyn, of the University of Manchester (U.K.), who helps students in teacher preparations programs to use telecommunications to discuss current issues in education and collaboratively prepare content-specific resources, instructional units, and problem solving simulations for local school children;

David Bell, of Simon Fraser University (Canada), who teachers preparatory courses both online and in person, and coordinates the SFU Xchange, a telecommunications network of preservice teachers, teachers, and university personnel;

Judi Harris, of the University of Virginia (U.S.A), who coordinates a teacher telecommunications network with which preservice teachers, public school teachers, school administrators, university professors, graduate students in education and public school students collaboratively explore educational and instructional issues and activities.

Panelists

Rhys Gwyn
University of Manchester
United Kingdom

David Bell
Simon Fraser University
Canada

Judi Harris
University of Virginia

Sponsor: ISTE

A Study of the Use of Interactive Videodisc Technology to Present Aural Tests to College Music Appreciation Students

Ernest Woodruff
Music Department
(816) 562-1317

Phillip Heeler
Computer Science Department
Northwest Missouri State University
Maryville, MO 64468
(816) 562-1600

Abstract

The effect of the use of interactive videodisc technology to present aural tests was studied in the context of college music appreciation classes. Both experimental and control groups were given study guides that identified the aural objectives for each test and specified the location of the musical examples that were to be used to illustrate the objectives. The experimental group was required to take aural tests over each unit in a supervised computer laboratory where a workstation consisting of a VT125 computer terminal connected to a Pioneer LDV-6000A videodisc player was housed. The control group did not take the computer-administered aural tests. The effect of the computer-administered aural tests was measured by scores on unit exams taken in the classroom. A significant superiority was found in the scores on unit exams taken by the experimental group ($p < .05$). It was concluded that under the conditions of the experiment, there was a significant advantage in requiring the taking of the computer-administered tests.

Introduction

According to Reimer (1989, p. 128), a respected philosopher of music education, a musical experience is one in which the listener perceives tonal relationships —melody, rhythm, etc.—and then reacts to their expressiveness. If one accepts this premise, it is obvious that a necessary part of the experience of music relates to the ability to perceive tonal relationships in the music. Therefore, an obvious goal of a music educator is to facilitate the improvement of listening perception by choosing music of quality for study and by using pedagogical techniques that focus on the experience of music.

Not only are the classroom techniques crucial to a student's improvement of listening skills, but listening outside class is necessary to make noticeable progress in the tonal comprehension of musical works. One of the major points made by Danziger (1984) is simply that repeated listening is how we make complex works more comprehensible.

Although it is quite common for teachers to expect outside listening, students often find that they are not successful with self-directed listening practice. It is also common to find that students will simply not take the time or make an effort to listen at all. The two-fold problem in this crucial area is (a) to provide students with a structured listening program that they can utilize on their own and (b) to structure the course in such a way that outside listening is rewarded.

Psychologist Fred Keller (1968), after many years as a college teacher, became dissatisfied with conventional teaching approaches. As a result of his training in psychology it was natural for him to develop a means of applying reinforcement theory to the teaching process. The Personalized System of Instruction (PSI) model, which Keller developed is one that addresses both of the problem areas mentioned above. According to Ryan (1974, p. 3) the behavior-analysis elements of this teaching technique are found in giving close attention to two very important but often relatively ignored aspects of teaching which are (a) clearly describing what is to be learned and (b) effectively managing the reinforcement of study. PSI has been used successfully in the sciences (Kulik, Kulik, & Carmichael, 1974), and it is recommended by Greer (1980) for all types of music classes.

PSI is a competency-based approach that requires the devision of a course into a number of manageable units or modules. The students are given specific objectives over these modules and are expected to master the content objectives at a predetermined criterion score. The format is individualized in that the students take the tests when they are ready and retake the tests until a criterion score such as 85% is achieved.

It is soon apparent to the would be user of a PSI model that a great amount of course preparation and proctoring of tests is required. This factor has undoubtedly kept many from adopting PSI even though it has a very favorable record of success. Much of the demand placed on the teacher using PSI can be reduced by using computer technology since the computer can tirelessly present multiple versions of tests, score them, and record the results.

The Electronic Campus project at Northwest Missouri State University has produced an environment which facilitates a computer application of PSI. At Northwest, every student residence hall room and faculty office has been equipped with a computer terminal. This terminal is connected to a campus wide network supported by several Digital Equipment Corporation VAX computers. The use of this mainframe system to test strictly informational objectives in a PSI context was a fairly simple procedure. However, the testing of aural (listening) objectives, which involved presenting musical excerpts and questions over them, presented a challenge to the researchers. Since the VAX is incapable of presenting acceptable music quality, a peripheral device was sought to serve this need.

Bork (cited in Lambert and Sallis, 1987) proposed the use of an intelligent videodisc system "where real phenomena are essential for understanding" a given process (p. 22). In this study, although the video component of an interactive videodisc system was unnecessary, the high quality audio available was well-suited to the needs of this project. The computer program mentioned above and previously used to administer strictly informational questions was modified to allow control of the videodisc player. In this program after a question was presented to the student, the program caused the videodisc player to search for and play the appropriate musical example. The random access feature of the videodisc system permitted multiple versions of tests to be generated by the computer. Without this type of system, an instructor would have to physically prepare multiple versions of a test, prepare separate audio recordings for each one, and then hand grade them. The latter alternative is not only less attractive to the instructor but also more restrictive to the students who would be able to take the tests only when test proctors were available.

A CD-ROM player with audio outputs would have been a logical choice for the study, but we found that access to this type of player was limited. In addition, the cost of the production of a compact disc with the necessary musical examples was much higher than that of a videodisc. It was found that a single videodisc using the recordable lasar videodisc technology could be produced for approximately $320 whereas the production of a single compact disc cost approximately $1500. As a result of these problems with CD-ROM, the videodisc format was chosen.

The purpose of this study was to determine the effect of computer generated aural tests in the setting of college music appreciation classes. A videodisc was produced containing relevant musical excerpts in order to conduct this study. The videodisc player was controlled by a computer program that randomly accessed musical examples correlated with teacher-prepared questions. The program selected one question for each of the aural objectives which were given to students as study guides. As students completed testing sessions the computer program gave them immediate feedback and recorded the results for the investigators.

Method

Subjects

Two sections of music appreciation students who were non-music majors at Northwest Missouri State University were available for a study of the use of interactive videodisc technology to present aural tests. The subject matter specialist taught both sections of music appreciation that were used in the study, and the computer specialist designed the computer program that administered the tests.

Design

Section three of Music 201 which met at 9:00 a.m. on Monday, Wednesday, and Friday was designated the experimental group, and section five, which met at 2:00 p.m. on the same days was designated the control group. Since intact groups were used for the investigation, the two sections were compared on the basis of ACT scores and years of previous musical experience (many students were first semester freshmen and therefore no college G.P.A. was available). A *t*-test was performed on the mean ACT scores and mean number of years of previous musical experience in order to determine whether there were significant differences between the two groups.

Both experimental and control groups were given study guides that identified aural objectives and specified the location of the musical examples for study in a record collection. The experimental group was required to take aural tests over each unit in a supervised computer laboratory. The workstation where the tests were administered consisted of a VT125 terminal with a Pioneer LDV-6000A videodisc player connected to it. Headphones were required for use by the students so that the musical examples would not disturb others who were using the same area. The control group did not take the computer-administered aural tests.

The effect of requiring the computer-administered auraltests was measured by the students' performance on five unit exams taken in class. The unit exams were based on the same objectives as the aural tests which were available to the experimental group on the computer. The mean scores on each of these unit exams were compared with a *t*-test.

Materials

The coursework for this study divided the aural objectives into 5 modules. Each module test had to be passed at the 85% level by midnight of the scheduled deadline before credit was awarded to the student. A total of 10% of the course grade was based on the timely completion of these module tests. Only one specially equipped terminal linked to a VAX 785 computer was available to administer the aural tests so the students reserved times to use the workstation.

The computer program was designed to meet several requirements. Firstly, it had to be able to generate a different version of a module test each time a student attempted it, and secondly, it had to produce one and only one question for each objective of the module test. The final task for the program was to access the appropriate musical example on the videodisc and play it when the student was ready to hear it. These requirements were achieved in a five hundred-line program written in VAX BASIC. Separate files for objectives and test items were created for each module. When the program was executed, students entered their social security number and selected the module number they wished to study. The program then read the correct objective file and loaded one randomly selected question for each objective into a large virtual array. The matching question file contained text, four alternative answers, and starting and stopping frame numbers for each question.

After the question array was built, the student was presented a question chosen at random from the array. The program played, on cue from the student, the indicated frames on the videodisc which contained the appropriate musical example. After the musical selection was played, the student was asked to select the correct answer for the question. The program then judged the response and scored it appropriately.

Other requirements for the program related to reporting the results of each attempted test. It was suggested by Hermann (1982) that students using computers as test administrators could do so successfully if they obtained proper feedback; therefore, at the completion of a module test the student's score, the objectives missed, and the location of the missed objectives in the study guide were displayed for the students. The student's score, the time, and date of the attempt were written to an indexed file from which a report could be generated to provide the investigators with the necessary information for the awarding of credit to each student.

Results

Students in this study were not randomly assigned to their groups; therefore, the two groups were tested for similarity on the basis of mean ACT scores and years of previous musical experience. The mean ACT scores for both groups were 19.64 (experimental) and 17.67 (control), and the standard deviations were 5.25 and 5.35 respectively. Although this is a sizeable difference, the results of the *t*-test indicated no significant difference ($p < .05$) between the two mean scores, $t(52) = 1.2305$, $p > 0.2241$.

The mean years of previous musical experience were 3.97 for the experimental group and 4.25 for the control group. The standard deviations were 4.14 and 4.03 for each group respectively. The *t*-test indicated no significant difference in the mean years of previous musical experience, $t(57) = 0.2360$, $p > 0.8143$.

At least on these two important variables the two groups were comparable. Therefore, it was assumed that a valid comparison of performance on exams could be made.

Mean scores for each of the unit exams is given for both groups in Table 1. Due to uncontrollable enrollment factors, the two groups differed considerably in size; however, the results of the *t*-test comparing all five unit test means were significant. Because the variances differed significantly on exam one, the data generated by using an unequal variance procedure was used. The data for exams two through five were based on the assumption of equal variance (Cody and Smith, 1987, p. 95).

Discussion

The investigators sought to determine whether or not the taking of computer-administered aural tests using interactive videodisc technology would influence unit exam scores. Traditionally students do worse on the aural portion of tests in a music appreciation class than they do on the portion that addresses the comprehension of informational objectives. The investigators expected that this testing program using interactive videodisc technology would create an increased awareness in students of their level of mastery of aural objectives

Summary of *t*-test Results Comparing the Mean Scores on Two Unit Exams of the Control Group (1) and Experimental Group (2)

Exam	Group	N	Mean	Standard Deviation	*t*	df	*p*
1	1	18	65.94	19.52			
	2	41	77.66	10.25	2.40	21.2	.025*
2	1	18	65.89	12.97			
	2	41	78.39	15.18	3.04	57.0	.004*
3	1	18	48.22	23.49			
	2	40	72.50	24.76	3.50	56.0	.0009*
4	1	18	54.72	24.22			
	2	40	69.83	20.87	2.42	56.0	.019*
5	1	18	75.05	14.85			
	2	39	85.49	12.30	2.78	55.0	.007*

*Note. These results are significant at the $p < .05$ level.

Table 1.

thereby causing them to prepare more effectively for unit exams taken in class. The data generated by this study did in fact suggest that students in the experimental group were better prepared for unit exams.

This study did not attempt to establish the validity of PSI as an instructional approach since many studies have provided sufficient evidence of its value. This study did provide data to support its use in encouraging the development of listening skills using the medium of interactive videodisc. Further, the study indicated that interactive videodisc was an expedient means of aural test administration. Its advantages are that it can facilitate the PSI approach by administering multiple test versions, giving the students feedback, and recording the results for the instructor. Therefore, the instructor is relieved of much of the labor involved in preparing and scoring tests.

While the testing program was successful, some problems arose which suggested further study. It was found that the videodisc system was not as user-friendly as might be desirable. Students often found that they needed help when communication problems developed between the videodisc player and the computer terminal. A more reliable system needs to be developed.

A second area for further study was suggested by the number of attempts a student required to pass a module test. It was found that some of the students attempted the tests as many as seven times without achieving a passing grade. With some students requiring many repetitions in order to pass a test, the investigators believe that interactive optical disc technology has great potential for the development of tutorials. For example, a CD-ROM player connected to a computer terminal could utilize commercially available compact discs to train students to hear features contained in the musical works before they attempted to take a test. This would be of particular help to students who are unable to distinguish musical features on their own.

The investigators credit the increased achievement of the experimental group in this study to the instructional design of PSI. The aural objectives were clearly specified, students were required to pass a test over all the objectives at a high criterion level, and they were rewarded appropriately for timely completion of the aural tests. The investigators believe, however, that the contribution of the medium of interactive videodisc to this study is no less significant. Without the labor-saving feature of the computer-generated tests, it would be far more difficult to implement the successful instructional qualities of PSI in thedevelopment of aural skills in a music appreciation course.

References

Cody, R. P. and Smith, J. K. (1987). *Applied Statistics and the SAS Programming Language*. New York; Elsevier Science Publishing.

Danziger, R. (1984). *The musical ascent of Herman Being*. New York: Jordan Press.

DeBloois, M. L. (1982). *Vidoedisc/microcomputer Courseware Design*. Englewood Cliffs, NJ: Educational Technology Publications.

Greer, R. D. (1980). *Design for Music Learning*. New York: Teachers College Press.

Herman, T. F. (1982). Effective tutoring in a PSI course, person vs. computer. (Report No. CG 016 828). Guelph, Ontario, Canada: University of Guelph. (ERIC Document Reproduction Service No. ED 233 251)

Keller, F.S. (1968). Good-bye teacher, *Journal of Applied Behavior Analysis*. *1*(1), 79-89.

Kulik, J. A., Kulik, C. L., & Carmichael, K. (1974). The Keller plan in science teaching. *Science*. *183*, 379-383.

Lambert, S. and Sallis, J. (1987). *CD-I and Interactive Videodisc Technology*. Indianapolis, IN: H.W. Sams.

Reimer, B. (1989). *A Philosophy of Music Education*. Englewood Cliffs, NJ: Prentice-Hall.

Ryan, B.A. (1974). *PSI: Keller's Personalized System of Instruction: An Appraisal*. Washington, D.C.: American Psychological Association.

Using "Teaching Information Systems" to Reduce the Barriers between Teaching and Research.

Russell L. Shackelford
Atlanta Thoughtworks One
51 Clark Place, Whitesburg, GA 30185

Albert N. Badre
School of Information and Computer Science
Georgia Institure of Technology

Abstract

We believe that the vast majority of effort in educational computing has effectively bypassed the teacher. We estimate that more than 99% of educational software development has focused on products for student use. As a consequence, we know little about what can be accomplished when teachers are "turned loose" with information systems that allow them access to detailed information about the teaching and learning process as it occurs in their own classrooms.

In this paper, we present an overview of the use of a PC-based *Teaching Information System* in the discovery of a widespread "teaching error" and of effective solutions to the newly discovered problem. First, we will summarize the rationale behind the *Teaching Information System* approach to Educational Computing. Next, we compare this approach to the generic CAI approach. This is followed by a summary of the features and capabilities of the *Teaching Information System* itself. We then present an overview of the use of a *Teaching Information System* in an important piece of research. Finally, we address the implications of this approach for teaching practice and for educational computing as a whole.

History and Context

This approach grew out of research and practice in the *"Computer Supported Instruction (CSI) Project"* at Georgia Tech. The *CSI Project* was one of several large university projects supported by IBM ACIS in the mid-1980s. Many, if not most, of these university projects focused on the development of software teaching materials of the student-interactive CAI variety. In contrast, Georgia Tech opted to pursue research into the use of computers in support of the "instructional infrastructure" throughout the Arts and Sciences.

The explicit goal was to increase "instructional productivity," i.e., to achieve "more" teaching and learning with the same level of human resources, utilizing affordable, off-the-shelf products wherever possible. Analyses were performed of the teaching and learning activities that are invariant across both subject matter and grade level, and of the flow of instruction-related information. Based on analyses of both (a) instructional activities and (b) information flow, the *Computer Supported Instruction (CSI) model* was developed.

The *CSI model* calls for four basic components:

1. *Student Workstations*: ordinary PC's, available to students in PC labs, in their dorms, and/or in their homes. These machines are used by students in conjunction with:
   - Application programs: word processors, spreadsheets, programming languages, databases, etc. (off-the-shelf components).
   - Utility programs: outliners, thesauri, spelling-checkers, etc. (off-the-shelf components).
   - Software specific to various subject matter: CAI modules, simulations, etc. (custom components).
2. *Instructional Delivery and Administration (IDA) System*: a PC network appropriate to the academic environment. It serves as a "communications backbone" which expedites the flow of information between teacher and student. We benefit from the availability of a wealth of off-the-shelf PC networking products. However, nearly all of these products were designed for the needs of the workplace, not the needs of schools. A PC network appropriate for academic use requires certain specific capabilities frequently lacking in "office networks."
3. *Teaching Information Systems (TIS)*: software designed explicitly for use by the educator, running on a medium-range PC (i.e., an "AT" or "286" machine). It tailors powerful and well-known information processing techniques to the special requirements of the educational environment.
4. *Presentation Support System (PSS)*: software and hardware to support the integration of computer graphics with traditional verbal and blackboard presentation techniques. Both projection monitors and graphics software are readily available. The educational environment has special requirements viz.: (a) the sequencing of graphic images, and (b) the integration of such sequences with other teaching materials.

Of these four components, the *Teaching Information System (TIS)* will be the focus of this presentation. It poses the greatest challenge and need because:

- It is central to the improvement in instructional productivity. The *TIS* is a core component, that which allows certain systemic benefits to accrue from use of the other components. It ties the others together into an integrated instructional system.
- It is the only component that had to be devised "from scratch." For each of the other three

components, the marketplace provides a number of excellent and affordable products. They may not be ideal for academic use; often, they fall quite short. But they are nonetheless quite useful, usable, and available. While not perfect, they are far better than nothing. With respect to the *Teaching Information System*, there was nothing to build upon: no significant professional-level software packages existed that would perform the necessary tasks.

The Relationship Between the Teaching Information System Approach and CAI

The *Teaching Information Systems (TIS)* approach can perhaps be most easily characterized as being the complement to the generic CAI approach to Education Computing:

CAI Approach	*TIS* Approach
• is *student-centered.*	• is *teacher-centered.*
• relies on *student use* of the computer.	• relies on *teacher use* of the computer.
• uses the computer as a *decision maker* (the computer decides what to do next.)	• uses the computer as a *decision aid* (the teacher decides what to do, but is aided by the computer, which provides easy access to relevant information).
• features subject matter content that is *programmed into the software.*	• features subject matter content that is *under teacher control.*
• is based on applications of specific *instructional methods.*	• is based on well known *information processing* techniques.
• seeks to, in some sense, directly *teach the student.*	• seeks to *empower the teacher* (and the educational system) to do a better job.

Beyond their differences, these approaches have much in common. Both approaches utilize PC-based computation to:

- support individualized instruction;
- generate feedback to students, and document feedback occurrences;
- allow monitoring and tracking of student performance;
- support need-based intervention to improve teaching and learning effectiveness.

Both the *TIS* and CAI approaches are components of the *CSI model*, each with its own role. Each can function independently of the other. Indeed, each can be, and has been, implemented without the other. However, they can also be used in an integrated fashion, with CAI modules feeding appropriate data to the TIS and vice versa. The basic idea is that the Student PC Workstations are a chief source of data, whether from CAI modules or from other kinds of computer use, while the TIS is where that data is processed into useful information about the teaching and learning process. This information can then be used to help optimize the process.

The Teaching Information Systems Approach to Educational Computing

The *Teaching Information System* approach is based on three important insights:

1. Most of the work that has been done in the realm of educational computing has been focused on providing instructional service to the *clients* of the educational system, i.e, the students. There has been insufficient attention to supporting the educational *professional*, i.e., the teacher and supervisory staff. In this respect, education has been unique in its basic approach: it is the *only* profession known to us which has deployed its limited computing resources primarily for client use rather than for professional use.

   Experience in the CSI Project taught that (a) the most successful computer applications are those that provide some useful service to the teacher, (b) many computer applications designed to directly service students actually increase teacher workload, which strongly inhibits their use in the real world, and (c) certain crucial aspects of teaching activity, which are invariant across both subject matter and grade level, do not benefit from *any* professional-level systems which increase teaching efficiency or effectiveness (various "classroom management" tools were evaluated and were found lacking with respect to both capability and usefulness).

   In response to these factors, the *TIS* approach is intended to: (a) "do something for the teacher," (b) help alleviate, rather than aggravate, teacher workload problems, and (c) support certain crucial teaching activities which are both subject matter independent and grade level independent.

2. In the domains of science and technology, rapid change has become a fact of life. Often, textbooks, software, and other teaching materials are obsolete by the time they reach their audience. Similarly, we are constantly learning more about effective techniques for teaching and learning. Thus, with

respect to both subject matter and educational practice, we require systems that are inherently dynamic and readily adaptable to change. It is necessary to find alternatives to "hard-wired" or "programmed" learning systems in software or other media. The *TIS* approach is intended to be inherently dynamic and adaptable to (a) changes in subject matter knowledge and (b) advances in teaching theory and practice.

3. No system of any kind can achieve effective self-regulation without adequate feedback loops. School systems are, in significant ways, attempting to function with feedback systems that are antiquated. It is clear that such loops have become inefficient and ineffective. Thus, it is necessary to develop systems that will *allow* school systems to self-regulate and optimize multiple aspects of established teaching and learning practices. The *TIS* approach is intended to establish information management loops which allow the quick recognition, identification, and response to problems in (a) student learning performance, (b) teacher instructional effectiveness, and (c) supervisory impact and curriculum integration. The goal is effective *self-regulation and local optimization* of the instructional process.

The CSI Project identified four areas of crucial teacher participation (**Preparation, Evaluation, Intervention, and Progress Monitoring**) as areas where computer impact is most lacking. There were *no* adequate systems available to assist educators in optimizing these crucial and universal functions. The *TIS* approach calls for integrated software systems, designed explicitly to augment teacher effectiveness in these areas.

The capture of student performance data is prerequisite to positive impact in each of these four areas. Once appropriate data has been captured, it can be be processed and utilized in various ways; without such data, we have nothing to go on. It has long been obvious that teachers have access to a rich supply of information about the teaching and learning process. Submitted student work contains a wealth of information about student strengths and weaknesses. The challenge is to capture and effectively exploit this information.

Typically, teachers record only a single grade in response to a student's submission; the useful information about student competence is effectively lost. This is largely because teachers have no practical way to capture the important student competence information. Instead, teachers can do little more than (a) jot two or three words in the margins of the student's paper, and (b) record a grade in the gradebook. Without information systems that support broad-based data capture, teachers have little choice but to throw away the very information that is needed to help truly optimize the teaching and learning process.

The *TIS* approach features one central theme: effective management of the information that modern technology makes available. Essential TIS processes include (1) effective capture of the important student performance information that already flows through the hands of teachers; (2) processing of that information to allow intervention to help students; (3) processing of that same information to allow "teaching effectiveness" feedback to both the teacher and the system.

Features of the Teaching Information System

Crucial to the *TIS* approach is the existence of appropriate software which allows inexpensive desktop computers to serve as advanced Teaching Information Systems. Until recently, such software simply did not exist. The Mindsight Corporation has, however, developed and released *"OPTIMUS, The Teaching Information System,"* which was designed to satisfy the requirements of the CSI model. *OPTIMUS* is a "processing engine" which integrates (a) student work evaluation environment , (b) performance feedback development and delivery system, (c) automated database system, (d) intervention management system, (e) teaching plans and materials integration system, and (f) teaching effectiveness evaluation system. An additional component, (g) systemic effectiveness analysis system, is in development. The system as a whole is adaptable to any subject matter or grade level. It provides no subject matter content itself, nor does it assume computing expertise on the part of the user. It is designed for use by educators, requiring the availability of one computer per teacher; this provides benefits to systems that currently lack much in the way of hardware for student use. It is most powerful when used in conjunction with student use of PC's.

The intended effect of integrating the *TIS* into existing practice is three-fold:

(1) the student benefits from improved performance feedback and intervention, while the teacher spends less time grading student work; student weaknesses are automatically analyzed and intervention needs can be identified quickly, thus increasing opportunity for constructive intervention before it is "too late"; clusters of students who share weaknesses can be identified, allowing more efficient use of the limited teacher time available for intervention; students (and their parents) can routinely benefit from individualized competency-specific guidance as to where study is most needed.

(2) the teacher learns from his own experience by easily identifying areas of strong teaching effectiveness and areas where improvement is needed; teaching thus becomes a self-correcting activity wherein each occasion of teaching a course allows the teacher to (a) know where he is strong and where he is challenged, (b) "fine tune" his work, based on empirical information, by focusing on targeted problem areas, and (c)

systematically monitor the results of such refinement efforts;

(3) the educational system is able to deploy available resources to improve precisely those areas of teaching where weaknesses are evident; supervisory and other resources (i.e., university-level experts) can be utilized to solve specific teaching problems which affect teaching effectiveness; the various strengths and weaknesses of individual teachers can be identified, thus providing a basis for "strength sharing" procedures among school system personnel; specific system-wide problems can be identified, thus providing a basis for a "problem solving" approach to curriculum development and integration; the school system benefits from a practical means by which to disseminate optimized teaching materials throughout the system.

Using a Teaching Information System to Identify and Solve Teaching Problems

The *Teaching Information System* approach serves as a support device for educators, and does not focus on basic research in teaching and learning. Nonetheless, the system has the potential to "tear down" significant barriers that have traditionally separated teaching practice and basic educational research from one another.

For example, we refer to how we used the *TIS* to discover, clarify, and apparently solve, serious hidden problems in the traditional approaches to teaching computer programming and problem solving in the Pascal programming language (Shackelford, 1989). While the specifics of that study involved computer science education, the techniques used are not limited to technical subjects. On the contrary, the *techniques* are generalizable, and we are excited about the potential such techniques hold for an explosion of insightful research *from the classroom*.

Purpose of the Research

This is the story of a research project that found more than it was looking for. Our research goals were rather mundane. After developing what we believed was a coherent, sensible, and practical approach to the use of computers in instruction (the CSI model), we were faced with various questions about whether our thinking was correct. Was this approach workable? Or was it just another collection of "blue sky" ideas that *sounded good* but wouldn't amount to much?

We had ample evidence in support of our views that the *main focus* of educational computing should be student use of *application* software rather than *subject matter specific* software. We had achieved far greater utilization of resources this way: an average of 4 hours of usage per week per enrollment using application programs, versus only about 20 minutes per week for CAI programs. This was consistent with data from other sources. We found both teachers and students *using* the computer resources this way and *feeling* positive about it as well: they felt like they were "getting things done." In contrast, we found that the CAI efforts left most participants frustrated and dissatisfied: students found the modules to be boring and stilting, and teachers found the approach to be "more trouble than it's worth." There were notable exceptions to this, as certain subject matter software had a very positive impact, but these were the exceptions to the rule. In sum, we felt confident that the application software approach was the predominant "way to go," with CAI programs being appropriate in special cases.

We also had ample evidence that using PC networks was a success. Teachers reported that the biggest improvement associated with the *CSI Project* was that both they and students could "get more done" in the same amount of time. For example, English professors found that the combination of (a) student use of word processors (rather than pen and paper), and (b) networked communications, allowed them to achieve more revisions per paper. They received student files in machine-readable form, evaluated student work and made comments to students online, returned these edited files to students, and expected prompt revisions of student work. In this way, they were able to get quicker "turnaround" than was possible with pen-and-paper methods. Thus, students no longer just submitted a paper and waited for a grade. Instead, students were expected to make multiple revisions of their work, as many as required to "get it right." The use of word processors and electronic media were credited with making this possible. In sum, the approach of using (a) student PC's as workstations on which students did their work, (b) teacher PC's as workstations on which teachers evaluated student work, and (c) PC networks as media by which students submitted their work and received evaluations, seemed to be a clear "winner."

We did not, however, have experience with the *Teaching Information System* approach, due to the lack of appropriate software. We did know that the idea of using a *Grading Environment* appeared to be sound. The basic idea is one of using the computer to exploit the typicality of student errors: using the *TIS*, the teacher responds to an error (or strength!) in a student's work by creating a "feedback comment" in whatever detail and length seem appropriate; the *TIS* then stores this comment such that it is easily accessible (organized by "competency areas") and may be used again for other students by simply selecting it from a menu of such comments. Thus, the teacher is able to provide detailed feedback to students via a very fast and easy method. The goal is fivefold: (1) improve the quality of feedback to students, (2) reduce the time required to grade student work, (3) take advantage of the automatic database functions of the *TIS* to quickly identify students who are in need of special intervention and guidance, (4) take advantage of the same database to identify areas where problems are typical and thus where instructional content or methods might need to be modified, and (5) track student performance after such changes in instruction in order to gauge the effectiveness of those changes.

Studies involving both English and computer science courses confirmed that student performance calls for a relatively small number of feedback comments. In both studies, teachers and graders (including those who were skeptical of the concept) found that less than 100 comments were required to handle everything they found in student work for an entire course. The great majority of these comments were created during the first term of use. The resulting database of comments was refined over the second and third terms, at which point the database seemed to remain rather constant.

Remaining questions about the *TIS* were: (a) "Would teachers and graders find such capability useful?," (b) "Would computer-applied feedback messages have any impact on subsequent student performance?", and (c) "Would the performance tracking capabilities provided by the *TIS* be of any actual use in improving local instructional practice?"

Our specific research goal was modest. We set out to investigate question (b), above: "Would computer-applied feedback messages make any difference in subsequent student performance?" For various reasons, we chose to look at this question in computer science, specifically within an area of introductory instruction: the use of iteration constructs.

The Surprising Findings

In the research study, novice programming students were assigned to four groups such that each group was equal with respect to both SAT score and performance on a pretest. All subjects were then presented with an experimental task. Each of the four groups received different types of performance feedback in response to their performance on this task. Following the differential treatment of the four groups, a post-test was administered to all subjects. The findings supported our hypothesis, in that certain kinds of feedback showed themselves to be more effective than others.

In performing the study, we had used the *OPTIMUS Teaching Information System* to process the various data. We evaluated subject responses and administered the various feedback messages using the *OPTIMUS Grading Environment*. We did so because of the time-saving advantages it provided: *OPTIMUS* effectively automated most of the tedious work involved. With *OPTIMUS* compiling and organizing our data for us, we then used a utility program to "cut and paste" the data into a standard spreadsheet program. We then used the spreadsheet to perform the appropriate statistical tests. In effect, we spent only a handful of hours doing work which otherwise would have consumed several days.

Our hypothesis was confirmed, but this is not what was most interesting or valuable about the study. As a result of having used *OPTIMUS*, we automatically had all of the performance and feedback data available in the database. This meant that we could immediately use the graphing and reporting capabilities of *OPTIMUS* to look at our data in various ways. In addition, we could easily "cut and paste" our data to spreadsheet and database programs for more complex analysis.

These capabilities proved to be a major boon in ways that we had NOT anticipated. Using them, we were able to notice strong and important patterns in subject responses, patterns that went beyond the scope of our initial hypothesis. Analysis of these patterns lead us to discover what appear to be two major errors in how programming in Pascal is typically taught, errors that permeated *all* of the 28 textbooks that we examined! This finding is much more powerful than the issue we set out to investigate. It has implications that go far beyond the use of computers in the instructional process: it calls for revision of both textbook content and teaching practice in introductory computer science instruction.

The details of our findings are beyond the scope of this paper. Suffice it to say that it appears that computer science authors and educators have been teaching students certain practices which encourage them to fail! Our research not only identified these problems, but proposed promising (and empirically supported) solutions to them. We think this is particularly important because it shifts the focus in computer science educational research away from things that educators cannot realistically do much about, i.e., "change Pascal," to things that computer science educators everywhere *can* effect: improving the quality of instruction by augmenting existing systems through new insights. For a detailed account, see Shackelford & Badre, in press.

Implications of the Findings

We believe that we have quite literally stumbled onto something very important here: the use of PC-based *Teaching Information Systems* to facilitate basic research about the teaching of existing subject matter.

In particular, we believe it is noteworthy that our raw data was simply a compilation of data that is currently available to *anyone* who teaches an introductory programming course. We gathered data about nothing more than (a) the correctness of the subjects' submissions, and (b) the distribution of feedback comments. Our discovery occurred as a result of simply correlating certain feedback comments (those which were responses to certain stylistic and implementation decisions) with submission correctness. This is precisely the sort of data that routinely flows through the hands of *every teacher* of introductory programming.

In short, we used a *Teaching Information System* much as any teacher might, and we gathered the same sort of data that would be available to any teacher. What is unusual is that, by using a *Teaching Information System*, we had a practical mechanism for capturing both (a) detailed student performance data and (b) an objective account of the various things we noticed about each student's performance; we also had (c) convenient access to that data. Normally, teachers do not have mechanisms for either capturing or retrieving such information: the only data typically recorded is a single grade per assignment. The *TIS* made the capture of detailed data nearly trivial for us, much as it would for any teacher. With that data available and accessible, it was an easy matter to discern the clear and powerful trends that were in it *despite the fact* that we weren't originally looking for them.

As a consequence, we accomplished more than we set out to do, not only with respect to the research at hand, but also with respect to the use of the *Teaching Information System*. We had conceived of such a system as supporting only the *immediate* instructional environment. Our larger goal had been to explore the usefulness of such a system as a tool to (a) allow improved feedback to students, (b) provide innovative effectiveness feedback to the teacher, and (c) help the local school system identify both strength and problem areas. We had not anticipated the value of a *Teaching Information System* as a tool for (d) research that expands the "state of the art" with respect to basic instructional treatments. Yet this is precisely what we found it to be.

The implications of this for educational computing are significant. Our findings are a concrete example of how information that is available from the classroom can lead to significant findings about both (a) the weaknesses of established instructional treatments and techniques, and (b) solutions which improve the effectiveness of instruction.

We find this particularly exciting because it is an example of how computers might directly help the educator do a more constructive job. We are reminded of other domains of successful computer application, domains where the computer is used *by professionals* and is used to *help professionals* get more things accomplished. In domains where computing has had significant impact, one of the chief effects is that of tearing down barriers: barriers between available information and those who need it, barriers between the gathering of data and the use of data, and barriers between those who ask the "information questions" and those who answer them. One of the chief contributions of the much heralded "computer revolution" has been to put valuable information tools on the desk of the professional. We find these advances sorely lacking in educational computing, and believe that *Teaching Information Systems* may play a significant role in rectifying this deficiency.

Everybody wants classroom teachers to be competant, to take initiative, and to discover better ways of doing things. At the same time, we observe that a major thrusts of the CAI approach to educational computing is to transfer the job of material presentation, performance evaluation, and feedback generation from the teacher to the software author. While this is no doubt appropriate in certain circumstances, we are left with serious reservations about its effect on the role of the teacher. We want to see technology deployed in the interest of helping the teacher get stronger. We do not see CAI addressing this issue in any meaningful way.

We are encouraged and enthused about the potential of *Teaching Information Systems* as practical tools which empower classroom teachers in new ways. We now believe that such systems can do more than improve the quality of feedback, do more than help the teacher become more effective, do more than help local systems monitor their strengths and weaknesses. Through our study, we have come to understand how a *Teaching Information System* can also serve as a "processing engine" to help classroom teachers discover more about the mysteries of teaching and learning. We believe that our experience is just one example of how computers can help tear down the barriers between teaching and research. We encourage educators everywhere to explore the possibilities for themselves.

References

Shackelford, R.L. (1989, in press). The Impact of Construct Definition Feedback on Looping Strategy Selection and Program Correctness. *Dissertations Abstracts International*.

Shackelford, R.L. & Badre, A.N. (in press). Why can't smart students solve simple programming problems. *Communications of the ACM*.

W2-12 COMPUTER SCIENCE AND ENGINEERING CURRICULUM... (ACM/SIGCSE & IEEE-CS)

Computer Science and Engineering Curriculum Recommendations: A Report from the Joint ACM/IEEE-CS Task Force

Allen B. Tucker, Chair
Bowdoin College
Brunswick, ME

Abstract

In the Spring of 1988, ACM and formed a joint task force to develop a new model for undergraduate programs in the discipline of computing. The report from this task force is intended to provide guidelines and recommendations for undergraduate programs named "computer science" and other similar designations, in various academic settings.

The task force has initially concentrated on the common requirements for undergraduate programs, with the objective of defining a set of common requirements to support all programs in the discipline. The content of the common requirements has been motivated by considering the minimal body of knowledge and capabilities that every graduate of a computing program should have. The content includes not only technical subject matter, but also such topics as ethics and professionalism.

The task force has used the work of the Task Force on the Core of Computer Science (Denning, et al, C. ACM, January 1989) as a basis for defining the common requirements. The nine areas of computer science, as defined in the Core Task Force Report, have also been used for the model curriculum. The common requirements have been subdivided into topic modules, currently called *knowledge units*, that can be combined in various ways for teaching purposes. A collection of principles, concepts, and objectives that occur throughout the curriculum have also been determined, and these are called *recurring themes*.

Some sample curricula based on the common requirements have been developed. The sample curricula address programs in liberal arts, sciences, and engineering contexts, with varies objectives for the programs.

Panelists

Keith Barker
University of Connecticut
Storrs, CT

Doris K. Lidtke
Towson State University
Baltimore, MD

A. Joe Turner
Clemson University
Clemson, SC

Sponsor: ACM/SIGSCE and IEEE-CS

Magic Characters Game

John T. Taylor
Hillsborough Community College

Marcelle Bessman
Frostburg State University

Abstract

The presenters will demonstrate, by interaction with the audience, a keyboard character game written in Turbo Pascal 5.0 that will introduce how a computer works using the binary number system. The presenter acts as a human computer, processing up to eight bits of information to identify a selected number or keyboard character.

This game, in addition to five Magic Number Games, introduces the concept of how switches (computer circuits) identify an "a" from an "A" from an input device, such as a keyboard, and the relationship of ASCII code to the binary number system.

The program also has an interactive tutorial explanation and lesson that follows the keyboard character game (those characters with ASCII values of 32 to 127) that distinguish a bit from a byte and why number manipulation requires only a nibble in ASCII. The Magic Numbers Games also have interactive tutorials that teach place value and counting in the binary number system.

Graphic displays do not require special graphic boards. The program will operate on the lowest level monochrome IBM-compatible microcomputers with 5.25" 360K drives. However, an improved version of the tutorials utilizes the advantages of hypermedia technology (IBM's *Linkway* software), which requires a minimum of IBM-compatible CGA color graphics, mouse, 640K, and 3.5" drives.

Free copies of the 5.25" disk program version will be made available to audience participants for their own classroom use. *Linkway* run time versions of the tutorials and games will also be available by 3.5" disk exchange.

The Other Side: Desktop Publishing for At-Risk Students

Kathleen A. Sutphen
T. E. Matthews County Community School
Marysville, California

Abstract

The Other Side was made possible by an Equal Time Education Grant from the Apple Computer Corporation. Apple donated a Macintosh computer lab to our school for the express purpose of providing an avenue for at-risk students to increase their academic skills and computer literacy through creating and producing a community newsletter. As part of producing this periodical the students interview community leaders and newsmakers, write stories, design and layout pages, create all artwork including photography, and oversee production. In the process, the students have gained technical skills, as well as listening, verbal, and writing skills, that are helping them break cycles of low self-esteem, academic failure, and criminal actions.

In my presentation I will give a brief overview of our student population along with statistical information about the surrounding area. I will discuss how I implemented the project, including introduction of software and computer usage to the students. I will discuss our pitfalls and our successes and indicate our students' reactions to our project.

To augment my presentation I will show slides of the students working on *The Other Side*, and I will use an overhead with a PC Viewer and a Macintosh II to better illustrate desktop publishing skills our students have mastered.

Computers, Curricula, and Special Learners

Gary G. Bitter, Mary Hatfield, and Janie Wilson
Arizona State University

Ruthie Blankenbaker
Park Tudor School, Indianapolis, Indiana

Shelley B. Wepner
William Paterson College

Abstract

How can special learners become the beneficiaries of computer embedded learning environments? This session will summarize ways in which the computer can be used across curricular areas to support special students' needs. Guidelines for software selection will be offered in mathematics, reading, and writing. The adaptability of a K through 8 mathematics model, including commercially prepared CAI software and other tool applications, will be presented. Software selection considerations for reading and writing will be offered in terms of special learners' psychological, intellectual, and procedural needs. An organizational description of an adapted writing-as-process electronic teaching environment will address the *whys* and *hows* of using technology with special learners. The establishment of a microcomputer lab will be described in terms of special learners' needs.

Linking with Europe

Joanne Troutner
Tippecanoe School Corporation
Lafayette, Indiana

Abstract

"Linking with Europe" will discuss a project involving sixth-grade students who have developed *Linkway* folders on modern European countries dealing especially with the cultural aspects and current events. The project is a joint effort between the sixth-grade teacher, the resource teacher for the gifted program, and the district technology coordinator. The students will work in teams to do the research and develop the computer materials. The project is designed to look at the effectiveness of cooperative learning and the use of *Linkway* as a teaching tool. Students in the control class will take the same content test as those in the experimental classroom. Students in both classes will also answer a survey that deals with the subjective portions of the project. An additional purpose of the project is to discover what teaching strategies work best when guiding middle school students in the use of hypermedia.

345

Incentive Funding in Technology: The New Jersey Model for Stimulating Improvement in Undergraduate Education

James Kinnamon
New Jersey Department of Higher Education

Abstract

The New Jersey Department of Higher Education in 1984 introduced state-run competitive grants to assist the state's colleges and universities—four-year and two-year, public and private—in using computing to improve undergraduate education. Over that period, approximately 400 projects selected from over 2,000 applications have directly benefited from the presence of state-provided grant opportunities. As a result of this considerable scope of activity, some valuable lessons have been learned about using incentive funding for technology-based projects to improve education.

From the perspective of the funding agency, the grant process, although a challenge to manage, has yielded impressive results with significant impact on students, academic departments, and institutions. The operation of a state-run grants program demands a serious commitment of people and energy, reasonable guidelines for prospective applicants, a fair decision-making process, and an effective means to monitor expenditures and grant-funded activities. Creative approaches to teaching with technology, improved grantsmanship, and heightened respect and recognition for the grant recipient are among the benefits.

From the perspective of faculty, a state-supported grants program holds hope for making costly technology-based improvements in curricula and pedagogy. The planning of a project, writing the grant, and implementation demand hard work and commitment from the faculty. However, faculty members working as project directors see the direct impact of a grant on students, academic departments, institutions, and themselves. The rewards of successful grant seeking, insights into writing successful grants, and making the promises of a grant application a reality are highlighted by two project directors, one from an art department and another from a communication department.

The presentation should be helpful to individuals considering applying for grants to use technology to teach. Anyone, including faculty and administrators at colleges and K-12 schools and state education agency officials, interested in using incentive funding to improve education in their own state, should find the experiences and knowledge shared particularly useful.

The Computer, A Developing Child's Tool of Vision

Nancy Scali
Arroyo School
Ontario, California

Abstract

Visions of the future are shaped by the learning of today. Children's visions capture the essence of a world they know, think about, and interpret through imagery and understanding.

"The Computer, A Developing Child's Tool of Vision," is a special session that explores the communication, artistic expression, and pictorial interpretation of a child's visual thinking. it will demonstrate that many computer qualities are natural to a child's visual literacy acquisition. It will support the extensions computers provide children in the act of creating artworks.

In the Fall of 1987, Arroyo School received an Apple Equity Project Grant. Consequently, an Apple IIGS computer with expansion card, color monitor, and mouse was placed in the kindergarten classroom. Room arrangements did not change. The computer was stationed on a moveable cart that was placed in a security closet each evening and weekends. The computer was to function as an art tool and to serve as a vehicle for critical thinking and creative problem solving in the areas of mathematics and science. The computer was conducive to a learning center structure emphasizing cooperative learning in small partner groups. Frequently, learning partners assisted pairs and such group interaction facilitated social and educational development.

Presentation Format

Examples of student artwork will illustrate:

1. Positive cooperative learning activities
2. Integration of computer art within the curriculum
3. Quality visual art activities for the single computer classroom
4. Art as a form of visual expression and communication

Ideas for parent involvement and awareness of the value of the computer as a tool for visual thinking will be shared.

A mini gallery will exhibit examples of professional computer artworks that have provided a visual stimulus for students:

Observation
Organization
Drawing Techniques
Painting Qualities
Media Integration
Visual Experimentation

Projects featured in *The Writing Notebook* and Sunburst's SOLUTIONS will be on display.

The potential of computer imagery, visual patterning, visual extension, art, and design will be discussed and an "idea book" will be provided.

Tomorrow's Multimedia Technology in Today's Classroom Media Integration Centre

Brent Wilson and Dale Henderson
Carleton Roman Catholic School Board
Nepean, Ontario, Canada

Abstract

We've all heard about it and read about it—but can it really be done? Restructuring curriculum, integrating high technology, compacting curriculum—are these all myths or reality?

The Carleton Roman Catholic School Board, located in Nepean (Ottawa), Canada, is not only ushering in a new decade—with the introduction of its Media Integrated Curriculum, the Board is racing towards the year 2000. Focusing on English, mathematics, and science in grades 4 through 6 and grades 7, 8, 9 through 12/0AC, Media Integration enhances the way the student and teacher interact.

This curriculum project works by linking high technology equipment to curriculum material. It is a two-fold process that allows students to learn course material by completing assignments, independently and in small groups, using various equipment. Teachers can now assess and evaluate student needs through a powerful computer software program. The curriculum puts technology such as computers, interactive videos, calculators, language masters, CD ROM materials, laserdisks, tape recorders, and video cameras directly into the hands of students and teachers. It is a hands-on, active learning approach.

The first step is the development of skills-based curriculum modules that integrate state-of-the-art technology. Skills and knowledge objectives are identified and organized into scope and sequence charts. Based on these skills, assignment cards are developed that encourage cooperative, individual, and small group learning. To cultivate motivation and student interest, the assignment cards have been created at three levels: enrichment, grade level, and reinforcement. By dividing the classroom into a variety of centres and utilizing the assignments and the Instructional Management System, teachers can provide an active learning environment—an environment in which sophisticated technology and curriculum design address different styles and rates of learning.

The Instructional Management System is a powerful computer software program that provides the teacher with detailed information of each student's progress and indicates individual needs. This computer based assessment and diagnostic software package also suggests instructional activities teachers can employ to meet student needs. It generates reports including individual, group, and class profiles. The Instructional Management System incorporates some of the most powerful tools yet developed to assist educators.

This session details how one Canadian board is making technology work on behalf of the students and teachers by bringing together the best of curriculum, educational philosophy, technology, and pedagogy.

348

Implementation of a Biometry and Medical Computing Course

Leo M. Harvill
James H. Quillen College of Medicine

John H. Kalbfleisch
East Tennessee State University

Abstract

A two-semester course sequence titled Biometry and Medical Computing for graduate students in the biomedical sciences was implemented during 1988-89 with six students. There were 13 students enrolled during the first semester of 1989-90. The course is a required six-credit-hour service course. Generally, it consists of a two-hour lecture and a two-hour microcomputer laboratory each week. The course is primarily a course in statistics and the use of statistical software for understanding statistical concepts and analyzing research data. There are introductions to several other computing topics including word processing, using computers in teaching, simulations, analog to digital conversion, graphics, literature searching, and BASIC programming. Students are given specific assignments in the laboratory sessions. There are also three examinations each semester. The statistical software used in 1988-89 course was SPSS Studentware. This year the Minitab package for microcomputers is being used. The hardware in the laboratory consists of IBM PC microcomputers with 640K memory, two floppy disk drives (360K), and no hard disk. The instructors are statisticians who are knowledgeable in the use of microcomputers. They utilize guest instructors from several departments in the university as well.

On Beyond Word Processing: Computer Supported Writing in the Elementary School Classroom

D. Midian Kurland
Education Development Center, Inc.

Charles W. Fisher
University of Northern Colorado

Abstract

Word processors and other electronic writing tools burst onto the educational scene a decade ago with much fanfare and promise. However, it is evident by now that tools alone are not enough to transform writing. This presentation will report on the efforts of a group of elementary school teachers interested in computer supported writing who, with support from the Apple Classroom of Tomorrow (ACOT) project, engaged in a year-long electronic learning circle. The learning circle provided a forum for discussing issues pertaining to classroom writing and instruction. It allowed teachers to interact and argue with their peers as well as with academic researchers, software developers, and corporate representatives.

For this presentation, Dr. Kurland, the learning circle moderator, will report on how the teachers reacted to the concept of the electronic learning circle, and what was accomplished online over the course of the year. Dr. Fisher will report on patterns of interaction in the electronic learning circle, and how discussions online related to classroom practice in selected classrooms. Several of the teachers involved in the project will be available to discuss how they integrate computer based writing tools into their curriculum, and how their approach to writing has been changed by their use of technology and their interactions over the network.

Computer Animation Software Libraries

William J. Joel
Graphics Research Group

Abstract

In order for students to develop animation systems without reinventing appropriate lower level source code, it is important to have in place libraries of code that will perform these basic functions. Both a course in Computer Animation as well as Independent Studies at Marist College require the student to either develop or more often extend animation software packages. If students were required to create these systems from scratch, a single semester would provide an insufficient time frame. This project entails the creation of a set of libraries suitable for the generation of software for such animation systems as keyframe, parametric, goal-directed, etc. These libraries are to be used by students and faculty in both their coursework and independent studies. Included in these libraries are Procedures and Functions for such operations as rendering a 3-dimensional hierarchical model, compositing transformation matrices and all appropriate I/O requirements. The presentation includes a demonstration of sample programs written utilizing these libraries, based upon keyframe animation concepts. The libraries are written in Turbo Pascal version 5.0.

Narrative Engines: Modeling the Application of Literary Theory

Peter Havholm and Larry L. Stewart
The College of Wooster

Abstract

Narrative engines are the result of a novel approach to teaching literary criticism at The College of Wooster. In the belief that the approach could be adapted to a theory course in any humanistic or social science discipline, we will describe the course structure and demonstrate the engines students created. Because the engines are constructed in *HyperCard*, the technique is easily adaptable to any Macintosh environment.

Narrative engines are attempts to represent a literary theory's explanatory power. Crudely, if the theory is a good one, a model accurately embodying its principles will be able to generate interesting story skeletons in the same way that a good grammar of natural language can "generate" complex and natural strings of language.

Students confront theory in a detailed way as they work to abstract rules from it, which can in turn be programmed. The challenge of designing a computer program that actualizes a theory's prescriptions is a challenge in understanding the theory more than a computer programming challenge. Some extremely stimulating discussion occurs as English students work with computer science students to come up with a program design that will adequately reflect a theory's explanatory power.

Self-Made Database Projects for University Level Education

Mary G. Harris and Donald L. Pratt
Bloomsburg University

Abstract

This project shows how locally-made computer databases can put new life into *Education in an Urban Society* class. Data become far more meaningful as students learn to use library resources, newspapers, government publications, and in some cases the telephone to obtain information. Students become actively involved in locating data and in generating charts and graphs. Statistics take on new life for students when they become immersed in gathering data and creating charts.

Additional advantages included (1) using the education class as a model for cooperative learning, (2) promoting higher levels of cognitive functioning, (3) active learning, and (4) practice in developing research skills.

Dr. Harris will present the material that was generated by her classes. The effectiveness of this approach will be readily apparent.

This session parallels Dr. Donald Pratt's session entitled, "Self-Made Database Projects for the Secondary School" (W1-6).

Managing Campus-Wide Information Systems

Timothy J. Foley
Lehigh University

Abstract

The rapid expansion of communication via computer conferencing, electronic mail, computer bulletin boards, international networks, and other forms of electronic communication such as online surveys has created many new issues and problems for colleges and universities. Many colleges and universities have implemented information systems that allow students, faculty, and staff to communicate electronically. As these systems have been implemented, policies on information management are usually developed on an ad-hoc basis with little knowledge of the legal, political, and management issues related to information management.

Lehigh, having implemented a campus-wide information system with over 6,000 individuals (85%) of the campus voluntarily opening accounts, has faced many of these management issues. The system currently has over 250 campus services and information sources, and provides electronic mail delivery at approximately one million messages per year. The following issues relating to managing information technology will be discussed: information management, censorship, security, legal issues, encouraging innovative uses, usage patterns, and training.

Changing the Precalculus Curriculum using Mathematica

Beva Eastman
William Paterson College

Abstract

This presentation will outline a high school precalculus curriculum that integrates Mathematica. Several specific assignments will be discussed along with the implications about changes in the curriculum, the methods of teaching, and most importantly, student learning.

Mathematica, with its library of mathematical functions and capability for symbolic manipulation, allows for an entirely different relationship with mathematics in which students ask and solve their own questions instead of answering questions posed by a teacher or textbook. The library of mathematical functions allows students easily to explore and move around with mathematical relationships instead of the usual discrete linear learning that so often occurs in the classroom. The capability for symbolic manipulation allows students to concentrate on mathematical thinking rather than on algebraic operations. In addition, the numerical calculations executed by Mathematica create a solid base for sophisticated applications.

These capabilities of Mathematica mean that most of the problems in present precalculus texts can be solved simply by typing in a single command. Thus, the curriculum must now be rethought. Since students now can formulate and solve their own problems, the teacher becomes a co-learner with the student. This presentation will also outline possible directions for the precalculus curriculum as well as implications for teacher training.

The Evaluative Imaging of Mental Models: Visualizing Cognitive Reality

Chris Dede
University of Houston—Clear Lake

Abstract

Our culture has evolved ways of visually displaying a few types of complex mental models (e.g., architectural blueprints). However, fields such as technical training lack a way of imaging the full scope of knowledge and skills that a curriculum is to convey. As a result, instructional developers, teachers, students, and curriculum evaluators are in a situation comparable to an artist trying to paint when, at any given time, only a tiny fraction of the canvas can be seen. We can scan a curriculum in microscopic detail, but cannot step back to get a richly detailed image of the whole.

Visualization (the application of computer graphics to the problem of perceiving typically intangible information) can be used to create a virtual environment mapping cognitive reality. This presentation will describe work in progress on developing an imaging tool for displaying the mental model underlying a technical training curriculum. The design strategy uses a hypertext browser to empower semantic navigation (movement among different conceptual perspectives in a complex information space). Long-term, this research could lead to an interface architecture for artificial realities (computer maintained worlds).

This research is sponsored by the Air Force Human Resources Laboratory and by NASA's Johnson Space Center.

Rural Teachers a Year Later: Providing Continuing Support in Educational Computing

Diane McGrath
Kansas State University

Abstract

During the year 1988-89, the Department of Education funded a project to train rural teachers through a WEEA grant (Women's Educational Equity Act) . Its purpose was to develop the computer skills, awareness of equity issues, and equity strategies available to rural teachers in order to provide computing opportunities for rural girls as well as boys. Seventeen rural middle school and upper elementary teachers completed the project, which was taught through a combination of distance learning methods and face-to-face meetings.

The presentation focuses on the findings of a follow-up project with these teachers one year after the end of their training. During early 1990, teachers' continuing support needs were assessed. Teachers expressed most often the need for additional coursework, other teachers to collaborate with, help in integrating the technology into their classes, finding/evaluating software, ideas for projects, and a hotline for getting help with problems. An assessment of the various methods of meeting these needs will be discussed. This assessment will include: teachers' computer practices, attitudes toward the various types of training and support systems tried, perceived changes in their classrooms, and observations about equity in their classrooms. In addition, the original training materials will be evaluated.

The Use of Computer Assisted Instruction in High School Courses Offered via Satellite

Susan M. McClelland, Linda Bennett, and William D. Cole
University of Mississippi

Abstract

The Midlands Consortium, one of the four funded Star Schools Projects, is currently offering high school instructional programming in foreign languages, sciences, math, U.S. government, economics, and Basic English and Reading to underserved students in small or geographically remote schools through the use of satellites and other distance learning technologies. The programs are broadcast two to three days per week. On the "off" days, the students participate in computer assisted instruction, use a computer based voice recognition unit, and work independently or in small or large groups on class assignments. A teaching partner or facilitator is assigned to each of these satellite classes.

The Office of Distance Learning at The University of Mississippi, a member of the Midlands Consortium, installed satellite receive equipment at 80 schools across the state of Mississippi. During the first year of classes (1989-90), 58 secondary schools offered courses by satellite to a total of 742 students enrolled in 10 different courses.

We will present some of the findings of the Midlands Research and Evaluation Committee concerning how computers were used in these secondary satellite courses in Mississippi and their effect on student achievement. Pre- and posttest scores on a standardized achievement test, student and facilitator survey data, and course grade information will be examined and the role of computer based technology in enhancing student achievement in a satellite course will be explored in detail.

Educational Technology Instruction and Integration Program for School of Education Methodology Faculty

Nancy J. Martin and Susan M. Cooper-Shoup
California State University, San Bernardino

Abstract

This presentation describes the efforts of a program to integrate technology into methodology instruction at the preservice level. This program also targets the inservice efforts coordinated to prepare the methodology faculty for this integration of technology. The plan evolved from the enactment of AB-1681 in the state of California that required candidates for a clear teaching credential to complete coursework in computer education. The California Commission on Teacher Credentialing (CTC) defined the required computer competencies that candidates for the clear teaching credential must possess. The competencies were separated into two levels (I and II). The School of Education at California State University, San Bernardino received grant monies to facilitate the full implementation of this plan as well as to prepare the methodology faculty to integrate the use of technologies into their coursework.

This presentation will describe the needs assessment completed by those faculty members as well as the ongoing inservice efforts provided for the methodology faculty. Discussion of the competencies, inservice preparation, and implementation of the legislation will be explored.

A Tutorial Introduction to Neural Networks

A. Martin Wildberger, Presenter
General Physics Corporation
Columbia, MD

Abstract

This tutorial is a brief, elementary introduction to the general field of neural networks. No specific technical background is required for its understanding, although general "computer literacy" is assumed and some familiarity with linear mathematics and matrix operations is helpful. The presentation is intended to provide enough background in neural networks for those interested to begin to follow technical developments in this area. It includes the following specific areas:

1. The origin of artificial neurons as models of biological neurons leading to their employment as simple processing elements in parallel computing systems.
2. The meaning of training and *learning* in the context of artificial neural networks, and how this differs from conventional programming.
3. Comparison of neural networks with other modeling tools such as simulation and expert systems, and the implications of their differences in selecting appropriate applications.
4. A brief survey of some of the most prominent and representative neural network paradigms and their applications.
5. A comparison of some hardware and software tools available for neural network simulation and development.
6. Demonstrations of a few of the tools, paradigms, and applications described.

Application areas used as illustrations include data reduction, reliable communication, acoustic signal discrimination, and simulation modeling. In the case of simulations used for education and training, neural networks have potential for reducing the cost of operation by reducing the computer power requirements, and by providing automatic and immediate corrective feedback to the user.

Sponsor: SCSI

355

Interactive Escher—Introductory Transformational Geometry Concepts in a Museum Environment

Gerry Segal
Bank Street College of Education

Abstract

Learning environments other then the classroom allow for the relatively easy implementation of curricula innovations. Interactive Escher is a software program developed for the Children's Museum of Manhattan (CMOM). Based on the mathematical illustrations of the Dutch artist M. C. Escher, it was created to allow children ages 5 through adult to explore 2-dimensional transformational geometry: specifically, tesselation of glide reflected patterns on the 2-dimensional plane.

The program was developed by the author as an outgrowth of an interdisciplinary mathematics curriculum for elementary and middle school students. The curriculum takes an investigative approach to structures in space. It studies topics in art , dance, chemistry, physics, and biology.

The presentation will include a demonstration of the program, a discussion of the underlying mathematics concepts, a discussion of the challenges of access to an educational computer program in a museum setting and how to integrate the program into the classroom.

Educational Electronic Networks: In Theory and in Practice

Hugh Mehan, Chair
University of California, San Diego

Bertrum Bruce, Discussant
BBN

Al Rogers, Discussant
San Diego County Office of Education

Abstract

This symposium will present research on the uses of long-distance electronic networks in K-12 schools. The participants represent most of the major groups conducting such research, so this symposium will provide a way for NECC attendees to find out the latest research findings in this rapidly developing instructional use of computer technology.

The papers that will be presented in this session are:

"Learning in Electronic Networks"

James Levin, Michael Waugh, Haesun Kim, Cathy Thurston, and Gin-Fon Ju
University of Illinois

Barbara Brehm
Educational Service Center #13, Rantoul, Illinois

Naomi Miyake
Aoyama Gakuin Women's University, Tokyo, Japan

"Building Educational Communities on Electronic Networks: Theory Into Practice"

Margaret Riel
AT&T Long Distance Learning Network

"The World in the Classroom: Interacting with Data from Outside the School"

Denis Newman
BBN Systems and Technologies Corp.

"Opening the Gate: Using Telecommunications in Mentoring Relationships with Elementary Students"

Shelley Goldman
Institute for Research on Learning, Palo Alto, California

Seth Chaiklin
Teachers College

"Connecting Kids: Telecommunications in the Science Classroom"

Cecilia Lenk
Technical Education Research Centers

Computing, Intellectual Property and Education—What are the Limits? Where are the Problems?

Frank W. Connolly, Chair
The American University
Washington, DC

Abstract

The evolution of intellectual property concepts in the computer age has educators, lawmakers, and the software industry scrambling for position. This session will consist of three presentations and a question and answer period on issues, laws, customs and economics of intellectual property. The focus will be education at large—primary and secondary through university level, computer centers, libraries, resource rooms, classrooms, software users and creators.

Participants

Steve Gilbert
EDUCOM
Princeton, NJ

Peter Lyman
University of Southern California
Los Angeles, CA

Sponsor: EDUCOM